

Part I

Core Components of Intelligent Agents

Chapter 2

Cognition

Human cognition represents a sophisticated information processing system that enables perception, reasoning, and goal-directed behavior through the orchestrated operation of multiple specialized neural circuits [98]. This cognitive architecture operates through mental states, which serve as the foundation where learning and reasoning occur. The remarkable ability to process information across different levels of abstraction and adapt to novel situations is a crucial inspiration for LLM agents [27].

The cognitive system exhibits several fundamental architectural properties reflected in Figure 1.1. First, learning functions across different mental state spaces: it can occur holistically across frontal lobes (supporting executive control and cognition) and temporal lobes (responsible for language, memory, and auditory processing), or focus on specific aspects for targeted improvement as shown by the varied research levels in the figure. Second, reasoning emerges in distinct patterns: it can follow structured templates for systematic problem-solving supported by logical reasoning and cognitive flexibility in the frontal lobes, or appear in unstructured forms for flexible thinking, particularly evident in decision-making and executive control functions. Third, the system demonstrates remarkable adaptability, continuously updating its mental states through experience while leveraging both supervised feedback (as in adaptive error correction in the cerebellum) and unsupervised environmental statistics, reflected in the different exploration stages of various cognitive functions shown in the figure [99].

These cognitive processes are supported by a modular organization, composed of distinct but interconnected components that form a cohesive system [100]. These modules include perception systems that transform raw sensory data into meaningful representations, memory systems that provide the substrate for storing and retrieving information, world models that support future scenario simulation, reward signals that guide refinement of behavior through reinforcement, emotion systems that modulate attention and resource allocation, reasoning systems that formulate decisions, and action systems that translate decisions into environmental interactions.

While human cognition implements these properties through complex neural architectures shaped by evolution, LLM agents attempt to approximate similar functions using large-scale neural models and algorithmic techniques. Understanding this biological-artificial parallel is crucial for developing more capable agents [101], as it highlights both the achievements and limitations of current systems compared to human cognition. Significant differences remain in areas such as adaptability, generalization, and contextual understanding.

In this section, we first explore **Learning**, examining both the spaces where it occurs within mental states and the specific objectives it serves. Subsequently, we investigate **Reasoning**, analyzing both structured and unstructured approaches, before concluding with a dedicated exploration of planning capabilities as a special reasoning action.

2.1 Learning

Learning represents the fundamental process through which intelligent agents transform experiences into knowledge within their mental states. This transformation occurs across different cognitive spaces, from holistic updates across the full mental state to refinement of specific cognitive components. The scope of learning encompasses remarkable capacities that serve different objectives: enhancing perceptual understanding, improving reasoning capabilities, and developing richer world understanding.

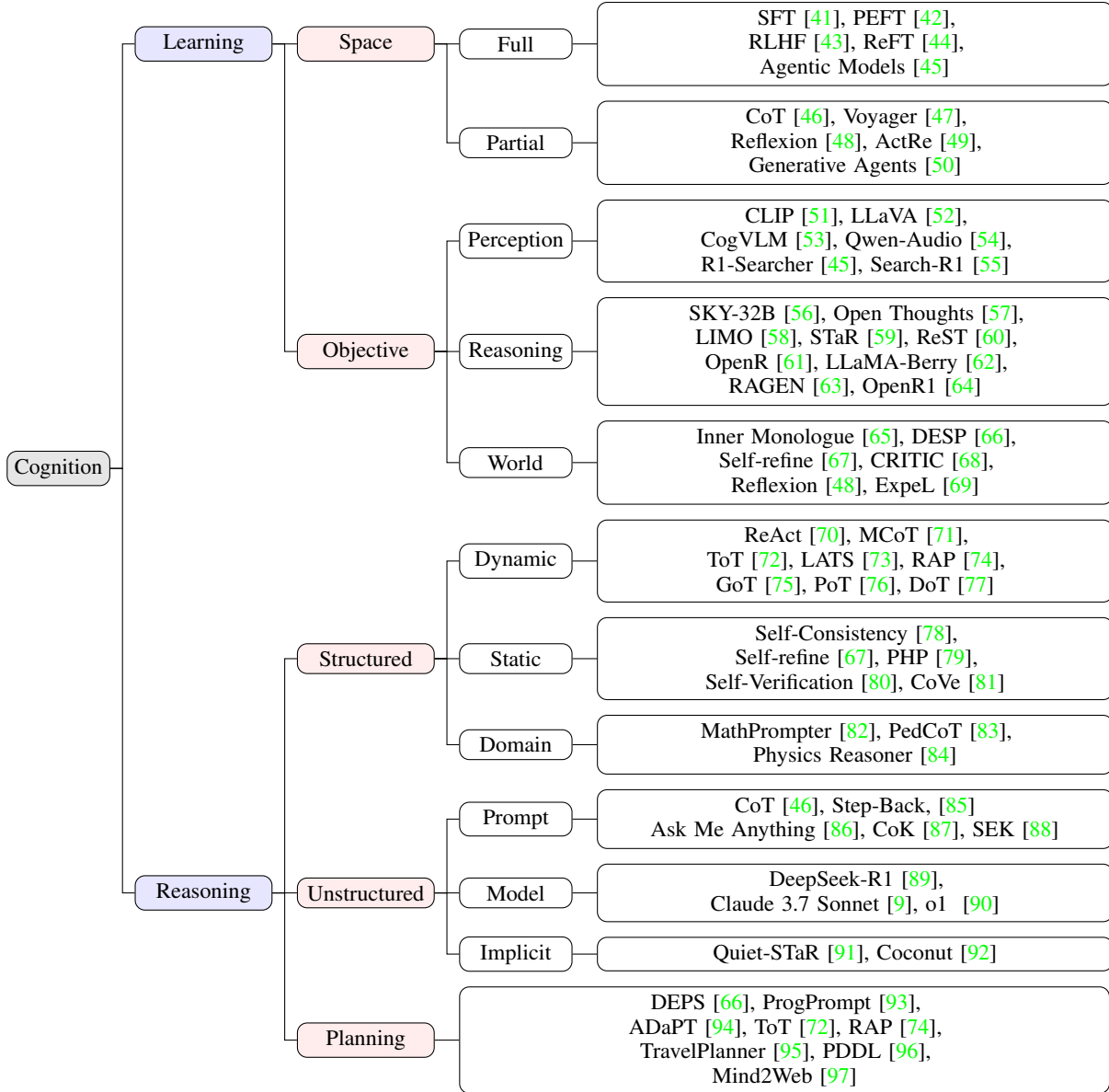


Figure 2.1: Illustrative Taxonomy of Cognition system, including learning and reasoning paradigm.

Human learning operates across multiple spaces and objectives through the brain’s adaptable neural networks. The brain coordinates learning across its entire network through integrated systems: the hippocampus facilitates rapid encoding of episodic experiences, the cerebellum supports supervised learning for precise motor skills, the basal ganglia enable reinforcement learning through dopaminergic reward signals, and cortical regions facilitate unsupervised pattern extraction [99]. At more focused levels, specific neural circuits can undergo targeted adaptation, allowing for specialized skill development and knowledge acquisition. These systems work together on different timescales, ranging from immediate responses to lifelong development, while being influenced by factors like attention, emotions, and social environment [27].

LLM agents, while fundamentally different in architecture, implement analogous learning processes across their mental state spaces. At the comprehensive level, they acquire broad knowledge through pre-training on massive datasets, demonstrating a form of unsupervised learning. At more focused levels, they refine specific capabilities through parameter-updating mechanisms like supervised fine-tuning and reinforcement learning. Uniquely, they also demonstrate in-context learning capabilities, adapting to novel tasks without parameter changes by leveraging context

within their attention window: a capability that mirrors aspects of human working memory but operates through fundamentally different mechanisms.

The comparison between human and artificial learning systems provides valuable insights for developing more capable, adaptive agents. Human learning demonstrates notable characteristics in efficiency, contextualization, and integration with emotional systems, while LLM-based approaches show distinct capabilities in processing large datasets, representing formal knowledge, and synthesizing information across domains. These complementary strengths suggest productive directions for research. As we explore the foundations of learning, we first examine the spaces where learning occurs within mental states, followed by an analysis of the specific objectives that drive learning processes.

Table 2.1: Summary of Learning Methods with Different State Modifications. ● indicates primary impact while ○ indicates secondary or no direct impact.

Method	Model	Perception	Reasoning	Memory	Reward	World Model
Voyager [47]	○	○	○	●	○	○
Generative Agents [50]	○	○	○	●	○	○
Learn-by-interact [102]	●	○	○	●	○	○
RAGEN [63]	●	○	●	○	●	○
DigiRL [103]	●	○	●	○	●	○
R1-Searcher [45]	●	●	●	○	●	○
RewardAgent [104]	●	○	○	○	●	○
Text2Reward [105]	○	○	○	○	●	○
ARAMP [106]	●	○	○	○	●	○
ActRe [49]	●	○	●	○	○	●
WebDreamer [107]	○	○	○	○	○	●
RAP [74]	○	○	○	○	○	●
AutoManual [108]	○	○	○	●	○	●

2.1.1 Learning Space

The learning approaches in LLM agents represent a structured, data-driven paradigm in contrast to the exploratory, emotionally-driven learning observed in humans. While human learning often involves active curiosity, motivation, and emotional reinforcement, LLM-based agents typically learn through more formalized processes, such as parameter updates during training or structured memory formation during exploration. Current agent architectures attempt to bridge this gap by implementing mechanisms that simulate aspects of human learning while leveraging the strengths of computational systems.

Learning within an intelligent agent occurs across different spaces, encompassing both the underlying model θ and mental states M , where the former fundamentally supports the capabilities and limitations of the latter. Formally, we define an intelligent agent’s internal state as a tuple $\mathcal{I} = (\theta, M)$ that includes both the model parameters and mental state components. The mental state can be further decomposed into different structures as we illustrated in 1.2:

$$M = \{M^{mem}, M^{wm}, M^{emo}, M^{goal}, M^{rew}\} \quad (2.1)$$

where M^{mem} represents memory, M^{wm} denotes world model, M^{emo} indicates emotional state, M^{goal} represents goals, and M^{rew} represents reward signals.

Modifications to the underlying model can be viewed as full mental state learning, as they fundamentally alter the agent’s capabilities. While model-level modifications can affect different mental states to varying degrees, changes to the model’s context window or external structures tend to focus on specific mental state components. For instance, learning experiences and skills from the environment primarily influence memory, while leveraging the LLM’s inherent predictive capabilities enhances the world model.

Full Mental State Learning Full mental state learning enhances the capabilities of an agent through comprehensive modifications to the underlying model θ , which in turn affects all components of the mental state M . This process begins with pre-training, which establishes the foundation of language models by acquiring vast world knowledge, analogous to how human babies absorb environmental information during development, though in a more structured and extensive manner.

Post-training techniques represent the cornerstone for advancing agent capabilities. Similar to how human brains are shaped by education, these techniques while affecting the entire model, can emphasize different aspects of cognitive

development. Specifically, various forms of tuning-based learning enable agents to acquire domain-specific knowledge and logical reasoning capabilities. Supervised Fine-Tuning (SFT) [41] serves as the fundamental approach where models learn from human-labeled examples, encoding knowledge directly into the model’s weights. For computational efficiency, Parameter-Efficient Fine-Tuning (PEFT) methods have emerged. Adapter-BERT [42] introduced modular designs that adapt models to downstream tasks without modifying all parameters, while Low-Rank Adaptation (LoRA) [109] achieves similar results by decomposing weight updates into low-rank matrices, adjusting only a small subset of effective parameters.

Some agent capabilities are closely connected to how well they align with human preferences, with alignment-based learning approaches modifying the model to reshape aspects of the agent’s underlying representations. Reinforcement learning from human feedback (RLHF) [110] aligns models with human values by training a reward model on comparative judgments and using this to guide policy optimization. InstructGPT [43] demonstrated how this approach could dramatically improve consistency with user intent across diverse tasks. Direct Preference Optimization (DPO) [111] has further simplified this process by reformulating it as direct preference learning without explicit reward modeling, maintaining alignment quality while reducing computational complexity.

Reinforcement learning (RL) presents a promising pathway for specialized learning in specific environments. RL has shown particular promise in enhancing reasoning capabilities, essentially enabling the agent’s underlying model to learn within the space of thought. Foundational works such as Reinforcement Fine-Tuning (ReFT) [44] enhance reasoning through fine-tuning with automatically sampled reasoning paths under online reinforcement learning rewards. DeepSeek-R1 [89] advances this approach through rule-based rewards and Group Relative Policy Optimization (GRPO) [112], while Kimi k1.5 [113] combines contextual reinforcement learning with optimized chain-of-thought techniques to improve both planning processes and inference efficiency. In specific environments, modifying models to enhance agents’ understanding of actions and external environments has proven effective, as demonstrated by DigiRL [103], which implements a two-stage reinforcement learning approach enabling agents to perform diverse commands on real-world Android device simulators.

Recent works have attempted to integrate agent action spaces directly into model training [45, 55], enabling learning of appropriate actions for different states through RL or SFT methods. This integration fundamentally affects the agent’s memory, reward understanding, and world model comprehension, pointing toward a promising direction for the emergence of agentic models.

Partial Mental State Learning While full mental state learning through model modifications provides comprehensive capability updates, learning focused on particular components of an agent’s mental state M represents another essential and often more efficient approach. Such partial mental state learning can be achieved either through targeted model updates or through in-context adaptation without parameter changes.

In-Context Learning (ICL) illustrates how agents can effectively modify specific mental state components without modifying the entire model. This mechanism allows agents to adapt to new tasks by leveraging examples or instructions within their context window, paralleling human working memory’s role in rapid task adaptation. Chain-of-Thought (CoT) [46] demonstrates the effectiveness of this approach, showing how agents can enhance specific cognitive capabilities while maintaining their base model parameters unchanged.

The feasibility of partial mental state learning is evidenced through various approaches targeting different components such as memory (M^{mem}), reward (M^{rew}), and world model (M^{wm}). Through normal communication and social interaction, Generative Agents [50] demonstrate how agents can accumulate and replay memories, extracting high-level insights to guide dynamic behavior planning. In environmental interaction scenarios, Voyager [47] showcases how agents can continuously update their skill library through direct engagement with the Minecraft environment, accumulating procedural knowledge without model retraining. Learn-by-Interact [102] further extends this approach by synthesizing experiential data through direct environmental interaction, eliminating the need for manual annotation or reinforcement learning frameworks. Additionally, agents can learn from their mistakes and improve through reflection, as demonstrated by Reflexion [48], which guides agents’ future thinking and actions by obtaining textual feedback from repeated trial and error experiences.

Modifications to reward and world models provide another example of partial mental state learning. ARMAP [106] refines environmental reward models by distilling them from agent action trajectories, providing a foundation for further learning. AutoMC [114] constructs dense reward models through environmental exploration to support agent behavior. Meanwhile, [107] explicitly leverages LLMs as world models to predict the impact of future actions, effectively modifying the agent’s world understanding (M^{wm}). ActRe[49] builds upon the language model’s inherent world understanding to construct tasks from trajectories, enhancing the agent’s capabilities as both a world model and reasoning engine through iterative training.

2.1.2 Learning Objective

The learning process of intelligent agents manifests across all aspects of their interaction with the environment. At the input level, agents learn to better perceive and parse environmental information; at the processing level, agents learn how to conduct effective reasoning based on existing knowledge or reasoning capabilities; at the comprehension level, agents form and optimize their understanding of the world through continuous interaction. This multi-level learning objective framework enables agents to evolve continuously across different dimensions, allowing them to better handle complex and dynamic task environments.

Learning for Better Perception The ability to effectively perceive and process information from the environment is fundamental to agent intelligence. To enhance perceptual capabilities, agents employ two primary learning approaches: expanding multimodal perception and leveraging retrieval mechanisms.

Multimodal perception learning enables agents to process and integrate diverse sensory inputs, similar to human multi-sensory integration but unconstrained by biological limitations. This capability has evolved significantly through advances like CLIP [51], which pioneered the alignment of visual and linguistic representations in shared embedding spaces. Building on this foundation, models like LLaVA [52] enhanced visual perception by training specialized projectors on image-text pairs, while CogVLM [53] advanced visual reasoning through unified representational architectures.

The expansion of perceptual modalities continues across multiple sensory domains. In audio processing, Qwen-Audio [54] demonstrates the unified encoding of diverse acoustic information, from speech to environmental sounds. Recent work by [115] has even ventured into tactile perception, developing datasets that align touch, vision, and language representations. These advances enable agents to engage more comprehensively with both physical and digital environments.

Agents also learn to enhance their observational capabilities through retrieval mechanisms. Unlike human perception, which is constrained by immediate sensory input, agents can learn to access and integrate information from vast external knowledge repositories. Retrieval-augmented approaches like RAG [116] enhance perceptual understanding by connecting immediate observations with relevant stored knowledge.

Recent work on retrieval-based agents demonstrates the potential for enhancing active information acquisition capabilities. Search-o1 [117] guides reasoning models to learn active retrieval through prompting, thereby expanding their knowledge boundaries. Taking this further, R1-Searcher [45] and Search-R1 [55] directly incorporate retrieval capabilities into the model, enabling autonomous information retrieval during the reasoning process. These advances suggest a promising direction for improving agent perception: enhancing model-level active perception capabilities to enrich the foundation for decision-making. This approach may represent a significant avenue for future agent development.

Learning for Better Reasoning Reasoning serves as a critical bridge between an agent’s mental state and its actions, making the ability to reason effectively and the development of reasoning capabilities essential for intelligent agents. The foundation of reasoning in modern agents stems from two key elements: the rich world knowledge embedded in their underlying models, and the robust logical frameworks supported either internally or through context structuring. This makes learning for better reasoning a vital objective in agent development.

The development of reasoning capabilities is demonstrated through several key phenomena. First, high-quality reasoning data directly enhances model reasoning ability; second, such high-quality data often requires verification or reward models for effective curation; and third, direct reinforcement learning on foundation models can spontaneously manifest reasoning capabilities.

The importance of reasoning in agent development has been re-emphasized following the release of the o1 series. A common approach involves collecting and distilling data from open/closed-source reasoning models. For instance, SKY-32B [56] distilled data from QWQ-32B [118] to train a 32B reasoning model at a cost of \$450. Similarly, Open Thoughts [57] trained Bespoke-Stratos-32B at a low cost by distilling and synthesizing datasets from R1. These studies demonstrate that even without complex algorithmic design, using reasoning data to perform Supervised Fine-Tuning (SFT) on base models can effectively activate reasoning capabilities.

Another crucial insight regarding data quality is that highly structured reasoning data more effectively enables agents and language models to learn reasoning processes. Notably, LIMO [58] demonstrated that powerful reasoning models could be built with extremely few data samples by constructing long and effective reasoning chains for complex reasoning tasks. This insight stems from their observation that language models inherently possess sufficient knowledge for reasoning but require high-quality reasoning paths to activate these capabilities. Supporting this view, Li et al.

[119] revealed that both Long CoT and Short CoT fundamentally teach models to learn reasoning structures rather than specific content, suggesting that automated selection of high-quality reasoning data may become an important future direction.

One viable exploration approach involves first conducting extensive searches, and then using verifiable environments or trainable reward models to provide feedback on reasoning trajectories, thereby filtering out high-quality reasoning data. This approach has led to several families of techniques that leverage different feedback mechanisms to improve reasoning capabilities.

The first category follows the bootstrap paradigm exemplified by STaR [59] and its variants, which implement techniques where models generate step-by-step rationales and iteratively improve through fine-tuning on successful reasoning paths. This family includes Quiet-STaR [91], V-STaR [120], and rStar-Math [121], with the latter specifically enhancing mathematical reasoning through reinforcement learning principles. By iteratively selecting correct reasoning paths for training, these methods achieve self-improvement through successive refinement cycles.

The second category extends this paradigm by more explicitly incorporating reinforcement learning principles. The ReST family, beginning with the original ReST [60] introducing reinforced self-training, performs multiple attempts (typically 10) per sample and creates new training datasets from successful reasoning instances. ReST-EM [122] enhances the approach with expectation maximization, while ReST-MCTS [122] further integrates Monte Carlo Tree Search to enable improved reasoning capabilities through more sophisticated exploration strategies.

Several approaches have introduced Policy Reward Models (PRMs) to provide quality feedback on reasoning paths. Methods like OpenR [61] and LLaMA-Berry [62] model reasoning tasks as Markov Decision Processes (MDPs) and leverage tree search to explore diverse reasoning paths while using PRMs for quality assessment. In domain-specific applications, methods like rStar-Math [121] and DeepSeekMath [112] have demonstrated success in mathematical problem-solving through multi-round self-iteration and balanced exploration-exploitation strategies. For code generation, o1-Coder [123] leverages MCTS to generate code with reasoning processes, while Marco-o1 [123] extends this approach to open-ended tasks. These implementations highlight how the synergy between MCTS and PRM achieves effective reasoning path exploration while maintaining solution quality through fine-grained supervision.

Beyond data-driven approaches, reinforcement learning (RL) has demonstrated remarkable success in enhancing language models' reasoning capabilities, as evidenced by recent breakthroughs like DeepSeek R1 [89] and Kimi-K1.5 [113]. The foundation of RL for LLMs can be traced to several pioneering frameworks: ReFT [44] introduced a combination of supervised fine-tuning and online reinforcement learning, while VeRL [124] established an open-source framework supporting various RL algorithms for large-scale models up to 70B parameters. RFT [125] further demonstrated the effectiveness of reward-guided optimization in specific reasoning tasks.

Building upon these foundations, subsequent works have explored diverse applications and improvements. OpenR1 [64] and RAGEN [63] extended RL techniques to enhance general reasoning capabilities, while specialized implementations like SWE-Gym [126] demonstrated success in software engineering tasks. Notably, DigiRL [103] introduced novel approaches for digital-world agent enhancement.

Recent advances have further integrated RL with tool usage and reasoning. Qwen-QwQ-32B [118] employs reinforcement learning and a general reward mechanism to incorporate tool calling into the reasoning process, enabling the seamless use of arbitrary tools during reasoning and achieving agent-like capabilities directly within the model. Similarly, RAGEN [63] focuses on multi-step agentic scenarios, establishing a framework for agent reinforcement learning in complex environments. These developments suggest an increasing convergence between model training and agent development, potentially leading to more integrated and capable intelligent systems. These implementations highlight how RL can effectively improve model performance while reducing dependence on large-scale annotated datasets, particularly in complex reasoning scenarios.

Learning for World Understanding A critical aspect of agent intelligence is the ability to understand how the world operates through direct interaction and experience accumulation. This understanding encompasses how the environment responds to different actions and the consequences these actions bring. Through continuous interaction with their environment, agents can build and refine their *memory*, *reward understanding*, and *world model*, learning from both successes and failures to develop a more comprehensive grasp of their operational domain.

Recent research has revealed diverse approaches to experiential learning for world understanding. At the foundational level, Inner Monologue [65] demonstrates how agents can accumulate basic environmental knowledge through continuous interaction. Similarly, Learn-by-Interact [102] shows that meaningful understanding can emerge from direct environmental engagement without explicit reward mechanisms. More sophisticated approaches are exemplified by DESP [66] and Voyager [47] in the Minecraft environment, where agents not only gather experiences but also actively process them: DESP through outcome analysis and Voyager through dynamic skill library expansion.

The processing and utilization of accumulated experiences have been further systematized through advanced frameworks. Generative Agents [50] introduces sophisticated memory replay mechanisms, enabling agents to extract high-level insights from past interactions. This systematic approach is enhanced by Self-refine [67] and Critic [68], which implement structured cycles of experience evaluation and refinement.

The optimization of reward understanding through environmental interaction has emerged as another crucial aspect of world understanding. Text2Reward [105] demonstrates how agents can continuously refine reward functions through human feedback, better aligning them with task objectives and environmental characteristics. Similarly, AutoManual [108] builds behavioral guidelines through sustained interaction, developing reward-verified protocols that provide a foundation for understanding environmental rewards and decision-making. These interaction-based optimization mechanisms enable agents to better comprehend environmental dynamics and generate more precise reward signals, ultimately enhancing their adaptability and decision-making capabilities in complex, dynamic environments.

Building on these foundations, RAP [74] represents a significant advancement by conceptualizing reasoning as planning with a world model. By repurposing LLMs as both reasoning agents and world models, RAP enables agents to simulate the outcomes of potential actions before committing to them, facilitating more effective planning through Monte Carlo Tree Search. This approach allows agents to strategically explore the reasoning space with a proper balance between exploration and exploitation.

Further innovations in leveraging world models for agent learning include ActRe [127], which reverses the typical reasoning-action sequence by first performing actions and then generating post-hoc explanations. This capability to rationalize actions demonstrates LLMs’ inherent understanding of world dynamics, enabling autonomous trajectory annotation and facilitating contrastive self-training.

The importance of cognitive maps in world understanding is highlighted by [128], who show that structured mental representations inspired by human cognition significantly enhance LLMs’ extrapolation capabilities in novel environments. These cognitive maps not only improve planning but also exhibit human-like characteristics such as structured mental simulation and rapid adaptation.

In web-based environments, recent work by [107] and [129] demonstrates that LLMs can function as effective world models for anticipating the outcomes of web interactions. By simulating potential state changes before executing actions, these approaches enable safer and more efficient decision-making, particularly in environments where actions may be irreversible.

Through systems like Reflexion [48] and ExpeL [69], agents have advanced experiential learning by autonomously managing the full cycle of experience collection, analysis, and application, enabling them to learn effectively from both successes and failures.

These developments collectively illustrate how world models are becoming increasingly central to agent learning systems, providing a foundation for understanding environmental dynamics and enabling more effective planning, reasoning, and decision-making in complex, interactive environments.

2.2 Reasoning

Reasoning represents the key to intelligent behavior, transforming raw information into actionable knowledge that drives problem-solving and decision-making. For both humans and artificial agents, it enables logical inference, hypothesis generation, and purposeful interaction with the world. In human cognition, reasoning emerges through multiple strategies: deductive reasoning applies general rules to specific cases, inductive reasoning builds generalizations from particular instances, and abductive reasoning constructs plausible explanations from incomplete data [130, 131]. These processes are augmented by heuristics—mental shortcuts that streamline decision-making under uncertainty—and are continuously refined through environmental feedback, ensuring that reasoning remains grounded in reality and adaptive to change.

For LLM-based agents, reasoning serves a parallel role, elevating them beyond reactive systems to proactive entities capable of sophisticated cognition. Through reasoning, these agents process multimodal inputs, integrate diverse knowledge sources, and formulate coherent strategies to achieve objectives. The environment plays a dual function: supplying information that fuels reasoning and serving as the proving ground where reasoned actions are tested, creating a feedback loop that enables agents to validate inferences and learn from errors.

In LLM-based agents, reasoning can be formally defined as the process of action selection based on mental states, representing a crucial bridge between perception and action. More precisely, given a mental state M_t at time t , reasoning can be formalized as a function $R(M_t) \rightarrow a_t$, where a_t represents the selected action. This process operates across

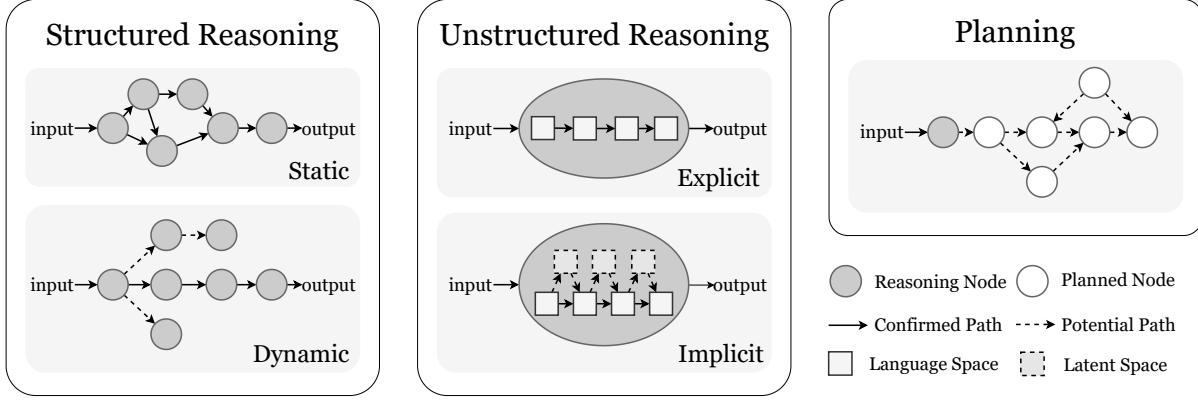


Figure 2.2: Comparison of reasoning paradigms in LLM-based agents.

various environments—textual, digital, and physical worlds—where completing a task typically requires either a single reasoning step or a composition of multiple reasoning actions.

The composition of reasoning actions naturally leads to two distinct approaches: structured and unstructured reasoning. Structured reasoning (R_s) can be formalized as an explicit composition $R_s = R_1 \circ R_2 \circ \dots \circ R_n$, where each R_i represents a discrete reasoning step with clear logical dependencies. In contrast, unstructured reasoning (R_u) takes a more holistic form $R_u = f(M_t)$, where the composition remains implicit and flexible, allowing for dynamic adaptation to context. This dual framework mirrors human cognition, where structured reasoning parallels our explicit logical deduction processes, while unstructured reasoning reflects our capacity for intuitive problem-solving and pattern recognition.

The environment plays a crucial role in this formalization, serving both as a source of observations o_t that influence mental state updates ($M_t = L(M_{t-1}, a_{t-1}, o_t)$) and as a testing ground for reasoning outcomes. This creates a continuous feedback loop where reasoning not only drives action selection but also influences how the agent’s mental state evolves, enabling iterative refinement of reasoning strategies through experience.

In this section, we will examine how these reasoning approaches manifest in practice. We begin with structured reasoning, which emphasizes systematic problem decomposition and multi-step logical chains. We then explore unstructured reasoning, which allows for flexible response patterns and parallel solution exploration. Finally, we investigate planning as a specialized form of reasoning that combines both structured and unstructured approaches for tackling complex, long-horizon tasks.

2.2.1 Structured Reasoning

Structured reasoning represents a methodical approach to problem-solving that employs explicit organizational frameworks to guide the reasoning process. Unlike unstructured approaches, structured reasoning makes the composition of reasoning steps explicit, which can be formalized as $R_s = R_1 \circ R_2 \circ \dots \circ R_n$, where each R_i represents a discrete reasoning step with clear logical dependencies. In this formulation, each reasoning node is an explicitly executed computational unit, and the connections between nodes represent definite information flow paths. This approach enables more systematic exploration of solution spaces and facilitates more robust decision-making through deliberate step-by-step analysis, providing high interpretability and traceability throughout the reasoning process.

2.2.1.1 Dynamic Reasoning Structures

Dynamic reasoning structures allow for the adaptive construction of reasoning paths during problem-solving, creating versatile frameworks that can adjust based on intermediate results and insights.

Linear Sequential Reasoning Linear structures frame reasoning as a series of sequential steps, where each step builds on the one before. ReAct [70] illustrates this by combining reasoning traces with task-specific actions in an alternating fashion. This combination allows for reasoning traces to guide and modify action plans while actions can access external sources for further information. This mutual interaction improves both reasoning integrity and environmental adaptation.

Reasoning via Planning (RAP) [74] extends the linear reasoning paradigm by formulating LLM reasoning as a Markov decision process, though it was limited by states specifically designed for particular problems. The Markov Chain of Thought (MCoT) [71] extended this paradigm by conceptualizing each reasoning step as a Markovian state accompanied by executable code. This approach enables efficient next-step inference without requiring a lengthy context window by compressing previous reasoning into a simplified math question. Atom of Thoughts [132] explicitly defined problems as state representations and designed a general decomposition-contraction two-phase state transition mechanism to construct Markovian reasoning processes, transforming complex problems into a series of atomic questions.

Tree-Based Exploration Tree-based approaches expand beyond linear structures by organizing reasoning into hierarchical frameworks that support branching exploration. Tree of Thoughts (ToT) [72] introduces a structured approach where complex problems are decomposed into intermediate steps, enabling breadth-first or depth-first search through the solution space. This allows the model to consider multiple reasoning paths simultaneously and systematically explore alternatives.

Language Agent Tree Search (LATS) [73] advances this paradigm by integrating Monte Carlo Tree Search (MCTS) with LLMs, using the environment as an external feedback mechanism. This approach enables more deliberate and adaptive problem-solving by balancing exploration and exploitation through a sophisticated search process guided by LLM-powered value functions and self-reflection.

Reasoning via Planning (RAP) [74] further enhances tree-based reasoning by repurposing LLMs as both reasoning agents and world models. Through this dual role, RAP enables agents to simulate the outcomes of potential reasoning paths before committing to them, creating a principled planning framework that balances exploration with exploitation in the reasoning space.

Graph-Based Reasoning Graph structures offer even greater flexibility by allowing non-hierarchical relationships between reasoning steps. Graph of Thoughts (GoT) [75] extends tree-based approaches to arbitrary graph structures, enabling more complex reasoning patterns that can capture interdependencies between different steps. This approach allows for connections between seemingly disparate reasoning branches, facilitating more nuanced exploration of the solution space.

Path of Thoughts (PoT) [76] addresses relation reasoning challenges by decomposing problems into three key stages: graph extraction, path identification, and reasoning. By explicitly extracting a task-agnostic graph that identifies entities, relations, and attributes within the problem context, PoT creates a structured representation that facilitates the identification of relevant reasoning chains, significantly improving performance on tasks requiring long reasoning chains.

Diagram of Thought (DoT) [77] models iterative reasoning as the construction of a directed acyclic graph (DAG), organizing propositions, critiques, refinements, and verifications into a unified structure. This approach preserves logical consistency while enabling the exploration of complex reasoning pathways, providing a theoretically sound framework grounded in Topos Theory.

2.2.1.2 Static Reasoning Structures

Static reasoning structures employ fixed frameworks that guide the reasoning process without dynamically adjusting the structure itself, focusing instead on improving the content within the established framework.

Ensemble Methods. Ensemble approaches leverage multiple independent reasoning attempts to improve overall performance through aggregation. Self-Consistency [78] pioneered this approach by sampling multiple reasoning paths rather than relying on single greedy decoding, significantly improving performance through majority voting among the generated solutions.

MedPrompt [133] demonstrates how domain-specific ensemble techniques can enhance performance by carefully crafting prompts that elicit diverse reasoning approaches, achieving state-of-the-art results on medical benchmarks through systematic composition of prompting strategies.

LLM-Blender [134] introduces a sophisticated ensembling framework that leverages the diverse strengths of multiple LLMs through pairwise comparison (PairRanker) and fusion (GenFuser) of candidate outputs. This approach enables the system to select the optimal model output for each specific example, creating responses that exceed the capabilities of any individual model.

Progressive Improvement. Progressive improvement frameworks focus on iteratively refining reasoning through structured feedback loops. Self-Refine [67] implements an iterative approach where the model generates initial output, provides self-feedback, and uses that feedback to refine itself. This mimics human revision processes without requiring additional training or reinforcement learning, resulting in significant improvements across diverse tasks.

Reflexion [48] extends this concept by integrating environmental feedback, enabling agents to verbally reflect on task feedback signals and maintain reflective text in an episodic memory buffer. This approach guides future decision-making by incorporating insights from previous attempts, significantly enhancing performance in sequential decision-making, coding, and reasoning tasks.

Progressive-Hint Prompting (PHP) [79] further develops this paradigm by using previously generated answers as hints to progressively guide the model toward correct solutions. This approach enables automatic multiple interactions between users and LLMs, resulting in significant accuracy improvements while maintaining high efficiency.

Error Correction. Error correction frameworks focus specifically on identifying and addressing mistakes in the reasoning process. Self-Verification [80] introduces a self-critique system that enables models to backward-verify their conclusions by taking the derived answer as a condition for solving the original problem, producing interpretable validation scores that guide answer selection.

Refiner [135] addresses the challenge of scattered key information by adaptively extracting query-relevant content and restructuring it based on interconnectedness, highlighting information distinction and effectively aligning downstream LLMs with the original context.

Chain-of-Verification (CoVe) [81] tackles factual hallucinations through a structured process where the model drafts an initial response, plans verification questions, independently answers those questions, and generates a final verified response. This deliberate verification process significantly reduces hallucinations across a variety of tasks.

Recursive Criticism and Improvement (RCI) [128] enables LLMs to execute computer tasks by recursively criticizing and improving their outputs, outperforming existing methods on the MiniWoB++ benchmark with only a handful of demonstrations per task and without task-specific reward functions.

Critic [68] extends this approach by integrating external tools for validation, enabling LLMs to evaluate and progressively amend their outputs like human interaction with tools. This framework allows initially “black box” models to engage in a continuous cycle of evaluation and refinement, consistently enhancing performance across diverse tasks.

2.2.1.3 Domain-Specific Reasoning Frameworks

Domain-specific reasoning frameworks adapt structured reasoning approaches to the unique requirements of particular domains, leveraging specialized knowledge and techniques to enhance performance in specific contexts.

MathPrompter [82] addresses arithmetic reasoning challenges by generating multiple algebraic expressions or Python functions to solve the same math problem in different ways. This approach improves confidence in the output results by providing multiple verification paths, significantly outperforming state-of-the-art methods on arithmetic benchmarks.

Physics Reasoner [84] addresses the unique challenges of physics problems through a knowledge-augmented framework that constructs a comprehensive formula set and employs detailed checklists to guide effective knowledge application. This three-stage approach—problem analysis, formula retrieval, and guided reasoning—significantly improves performance on physics benchmarks by mitigating issues of insufficient knowledge and incorrect application.

Pedagogical Chain-of-Thought (PedCoT) [83] leverages educational theory, particularly the Bloom Cognitive Model, to guide the identification of reasoning mistakes in mathematical contexts. This approach combines pedagogical principles for prompt design with a two-stage interaction process, providing a foundation for reliable mathematical mistake identification and automatic answer grading.

The evolution of structured reasoning in LLM agents reflects a growing understanding of how to enhance reasoning capabilities through explicit organizational frameworks. From linear sequences to complex graphs, and ensemble methods to specialized domain frameworks, these approaches demonstrate the power of structural guidance in improving reasoning performance across diverse tasks and domains.

2.2.2 Unstructured Reasoning

In contrast to structured reasoning approaches that explicitly organize reasoning steps, unstructured reasoning (R_u) takes a holistic form $R_u = f(M_t)$, where the composition remains implicit and flexible. In this mode, the reasoning process is encapsulated within a single function mapping, without explicitly defining intermediate steps or state transitions. This approach leverages the inherent capabilities of language models to generate coherent reasoning without enforcing rigid structural constraints, with intermediate reasoning processes occurring explicitly in the language space or implicitly in the latent space. Unstructured reasoning methods have demonstrated remarkable effectiveness across diverse tasks while maintaining simplicity and efficiency in implementation.

2.2.2.1 Prompting-Based Reasoning

The most accessible way to elicit reasoning in LLM agents lies in carefully crafted prompts. By providing appropriate reasoning demonstrations or instructing LLMs to perform inferential steps, agents can leverage their logical deduction capabilities to solve problems through flexible reasoning processes.

Chain-of-Thought Variants. The cornerstone of prompting-based reasoning is Chain-of-Thought (CoT) prompting [46], which operationalizes reasoning through few-shot examples with explicit generation of intermediate rationalization steps. This foundational technique has inspired several evolutionary variants that enhance its basic approach. Zero-shot CoT [136] eliminates the need for demonstration examples through strategic prompting (e.g., “Let’s think step by step”), making the approach more accessible while maintaining effectiveness. Auto-CoT [137] automates the creation of effective demonstrations by clustering diverse questions and generating reasoning chains for representative examples from each cluster. Least-to-Most Prompting [138] addresses complex reasoning by decomposing problems into sequential sub-problems, enabling a progressive planning process that facilitates easy-to-hard generalization. Complex CoT [139] further enhances reasoning depth by specifically selecting high-complexity exemplars as prompting templates, better-equipping models to tackle intricate problems.

Problem Reformulation Strategies. Advanced prompting strategies demonstrate architectural innovations in reasoning guidance by reformulating the original problem. Step-Back Prompting [85] implements abstraction-first reasoning through conceptual elevation, enabling models to derive high-level concepts and first principles before addressing specific details. Experimental results demonstrate substantial performance gains on various reasoning-intensive tasks, with improvements of 7-27% across physics, chemistry, and multi-hop reasoning benchmarks. Rephrase and Respond [140] employ semantic expansion to transform original questions into more tractable forms, allowing models to approach problems from multiple linguistic angles and identify the most effective problem formulation.

Abstraction-of-Thought [141] introduces a novel structured reasoning format that explicitly requires varying levels of abstraction within the reasoning process. This approach elicits language models to first contemplate at the abstract level before incorporating concrete details, a consideration overlooked by step-by-step CoT methods. By aligning models with the AoT format through finetuning on high-quality samples, the approach demonstrates substantial performance improvements across a wide range of reasoning tasks compared to CoT-aligned models.

Enhanced Prompting Frameworks. Several frameworks extend the basic prompting paradigm to create more sophisticated reasoning environments. Ask Me Anything [86] constrains open-ended generation by reformulating tasks into structured question-answer sequences, enforcing focused reasoning trajectories. This approach recursively uses the LLM itself to transform task inputs to the effective QA format, enabling open-source GPT-J-6B to match or exceed the performance of few-shot GPT3-175B on 15 of 20 popular benchmarks.

Algorithm of Thoughts [142] proposes a novel strategy that propels LLMs through algorithmic reasoning pathways by employing algorithmic examples fully in context. This approach exploits the innate recurrence dynamics of LLMs, expanding their idea exploration with merely one or a few queries. The technique outperforms earlier single-query methods and even more recent multi-query strategies while using significantly fewer tokens, suggesting that instructing an LLM using an algorithm can lead to performance surpassing that of the algorithm itself.

Chain-of-Knowledge (CoK) [87] augments LLMs by dynamically incorporating grounded information from heterogeneous sources, resulting in more factual rationales and reduced hallucination. CoK consists of three stages: reasoning preparation, dynamic knowledge adapting, and answer consolidation, leveraging both unstructured and structured knowledge sources through an adaptive query generator. This approach corrects rationales progressively using preceding corrected rationales, minimizing error propagation between reasoning steps.

Self-Explained Keywords (SEK) [88] addresses the challenge of low-frequency terms in code generation by extracting and explaining key terms in problem descriptions with the LLM itself and ranking them based on frequency. This approach significantly improves code generation performance across multiple benchmarks, enabling models to shift attention from low-frequency keywords to their corresponding high-frequency counterparts.

2.2.2.2 Reasoning Models

Recent advances in language models have led to the development of specialized reasoning models designed explicitly for complex inferential tasks. These models are fine-tuned or specially trained to optimize reasoning capabilities, incorporating architectural and training innovations that enhance their performance on tasks requiring multi-step logical inference.

Reasoning models like DeepSeek’s R1 [89], Anthropic’s Claude 3.7 Sonnet [9], and OpenAI’s o series models [90] represent the frontier of reasoning capabilities, demonstrating remarkable proficiency across diverse reasoning bench-

marks. These models are trained with specialized methodologies that emphasize reasoning patterns, often incorporating significant amounts of human feedback and reinforcement learning to enhance their inferential abilities.

The emergence of dedicated reasoning models reflects a growing understanding of the importance of reasoning capabilities in language models and the potential benefits of specialized training for these tasks. By concentrating on reasoning-focused training data and objectives, these models achieve performance levels that significantly exceed those of general-purpose language models, particularly on tasks that require complex logical inference, mathematical reasoning, and multi-step problem-solving.

2.2.2.3 Implicit Reasoning

Beyond explicit reasoning approaches, recent research has explored the potential of implicit reasoning methods that operate without overtly exposing the reasoning process. These approaches aim to improve efficiency by reducing the number of tokens generated while maintaining or enhancing reasoning performance.

Quiet-STaR [91] generalizes the Self-Taught Reasoner approach by teaching LMs to generate rationales at each token to explain the future text, improving their predictions. This approach addresses key challenges including computational cost, the initial unfamiliarity with generating internal thoughts, and the need to predict beyond individual tokens. Experimental results demonstrate zero-shot improvements in mathematical reasoning (5.9%→10.9%) and commonsense reasoning (36.3%→47.2%) after continued pretraining, marking a step toward LMs that learn to reason in a more general and scalable way.

Chain of Continuous Thought (Coconut) [92] introduces a paradigm that enables LLM reasoning in an unrestricted latent space instead of using natural language. By utilizing the last hidden state of the LLM as a representation of the reasoning state and feeding it back as the subsequent input embedding directly in the continuous space, Coconut demonstrates improved performance on reasoning tasks with fewer thinking tokens during inference. This approach leads to emergent advanced reasoning patterns, including the ability to encode multiple alternative next reasoning steps, allowing the model to perform a breadth-first search rather than committing to a single deterministic path.

Recent analysis [143] of implicit reasoning in transformers reveals important insights into its limitations. While language models can perform step-by-step reasoning and achieve high accuracy in both in-domain and out-of-domain tests via implicit reasoning when trained on fixed-pattern data, implicit reasoning abilities emerging from training on unfixed-pattern data tend to overfit specific patterns and fail to generalize further. These findings suggest that language models acquire implicit reasoning through shortcut learning, enabling strong performance on tasks with similar patterns while lacking broader generalization capabilities.

The evolution of unstructured reasoning approaches demonstrates the remarkable adaptability of language models to different reasoning paradigms. From simple prompting techniques to sophisticated implicit reasoning methods, these approaches leverage the inherent capabilities of LLMs to perform complex logical inferences without requiring explicit structural constraints. This flexibility enables more intuitive problem-solving while maintaining efficiency and effectiveness across diverse reasoning tasks.

2.2.3 Planning

Planning is a fundamental aspect of human cognition, enabling individuals to organize actions, anticipate outcomes, and achieve goals in complex, dynamic environments [144]. Formally, planning can be described as the process of constructing potential pathways from an initial state to a desired goal state, represented as $P : S_0 \rightarrow \{a_1, a_2, \dots, a_n\} \rightarrow S_g$, where S_0 is the starting state, $\{a_1, a_2, \dots, a_n\}$ denotes a sequence of possible actions, and S_g is the goal state. Unlike direct reasoning, planning involves generating hypothetical action sequences before execution, functioning as computational nodes that remain inactive until deployed. This cognitive ability emerges from the interplay of specialized neural circuits, including the prefrontal cortex, which governs executive control, and the hippocampus, which supports episodic foresight and spatial mapping. Insights from decision theory, psychology, and cybernetics—such as rational frameworks, prospect theory, and feedback loops—demonstrate how planning allows humans to transcend reactive behavior, actively shaping their futures through deliberate intent and adaptive strategies. This capacity not only underpins intelligent behavior but also serves as a model for developing LLM-based agents that seek to replicate and enhance these abilities computationally [145, 146].

In human cognition, planning operates as a hierarchical process, integrating immediate decisions with long-term objectives. This reflects the brain’s modular architecture, where neural systems collaborate to balance short-term demands with future possibilities—a dynamic informed by control theory’s principles of stability and optimization. Similarly, LLM-based agents employ planning by leveraging their extensive linguistic knowledge and contextual reasoning to transform inputs into actionable steps. Whether addressing structured tasks or unpredictable challenges,

these agents emulate human planning by decomposing objectives, evaluating potential outcomes, and refining their strategies—blending biological inspiration with artificial intelligence. This section examines the theoretical foundations and practical techniques of planning, from sequential approaches to parallel exploration, highlighting its critical role in intelligent systems.

Despite the potential of LLMs in automated planning, their performance faces limitations due to gaps in world knowledge [147]. LLMs often lack deep comprehension of world dynamics, relying on pattern recognition rather than genuine causal reasoning, which hinders their ability to manage sub-goal interactions and environmental changes [148]. Additionally, their reliance on static pre-training data restricts adaptability in real-time scenarios, limiting their generalization in dynamic planning tasks [149]. The absence of an intrinsic System 2 reasoning mechanism further complicates their ability to independently generate structured, optimal plans [150]. However, researchers have proposed strategies such as task decomposition, search optimization, and external knowledge integration to mitigate these challenges.

Task Decomposition Task decomposition enhances LLM planning by breaking complex goals into smaller, manageable subtasks, reducing problem complexity and improving systematic reasoning. The Least-to-Most Prompting method [138] exemplifies this approach, guiding LLMs to solve subproblems incrementally. ADaPT [151] further refines this strategy by dynamically adjusting task decomposition based on complexity and model capability, particularly in interactive decision-making scenarios. These methods also facilitate parallel subtask processing, backward error tracing, and independence determination [132], providing a structured framework for reasoning.

In LLM planning, tasks function as executable units—distinct from static state descriptions in formal models—emphasizing structured sequences that achieve intended outcomes [66]. These tasks vary in nature: some are subproblems requiring specific solutions (e.g., solving equations within broader challenges), while others involve tool invocation (e.g., querying APIs for weather data in travel planning) [152, 153]. Alternatively, tasks may be represented as graph nodes mapping dependencies, such as prioritizing objectives in project management [154]. By defining clear, modular goals, these formulations enhance reasoning and action efficiency, guiding agents through complex problem spaces with greater precision [93].

Searching Given the stochastic nature of LLMs [155], parallel sampling combined with aggregated reasoning can improve inference performance. Task decomposition structures individual solution trajectories, enabling the construction of a solution space that includes multiple pathways to a goal and their interrelationships [72, 156]. This space allows sampling diverse potential solutions [157], facilitating exploration through techniques like reflection, review, and parallel sampling informed by existing knowledge [158].

Computational constraints often preclude exhaustive evaluation, making efficient navigation of the solution space essential. Methods include tree search algorithms like LATS [159], heuristic approaches such as PlanCritic’s genetic algorithms [160], and CoT-SC, which identifies recurring solutions via self-consistency checks [78]. Reward-based models like ARMAP assess intermediate and final outcomes to optimize planning [106]. This iterative exploration and refinement process enhances adaptability, ensuring robust strategies for complex problems.

World Knowledge Effective planning requires agents to navigate dynamic environments, anticipate changes, and predict outcomes, underscoring the importance of world knowledge. RAP [74] examines the interplay between LLMs, agent systems, and world models, positioning LLMs as dual-purpose entities: as world models, they predict state changes following actions [107, 161]; as agents, they select actions based on states and goals [70]. This framework mirrors human cognition—simulating action consequences before selecting optimal paths—and unifies language models, agent models, and world models as pillars of machine reasoning [162].

Agents augment LLM capabilities by integrating external knowledge, addressing gaps in world understanding. ReAct employs an action-observation loop to gather environmental feedback, combining real-time data with linguistic knowledge to improve decision-making in complex scenarios [70]. This enables LLMs to iteratively refine their world models during action execution, supporting adaptive planning. Conversely, LLM+P [163] integrates LLMs with the PDDL planning language, converting natural language inputs into formalized representations solved by classical planners [164, 165]. This hybrid approach compensates for LLMs’ limitations in structured planning, merging their linguistic flexibility with the reliability of traditional systems.

Further advancements enhance LLM planning through world knowledge integration. CodePlan [166] uses code-form plans—pseudocode outlining logical steps—to guide LLMs through complex tasks, achieving notable performance improvements across benchmarks [167]. The World Knowledge Model (WKM) equips LLMs with prior task knowledge and dynamic state awareness, reducing trial-and-error and hallucinations in simulated environments [168]. A neuro-symbolic approach combining Linear Temporal Logic with Natural Language (LTL-NL) integrates formal logic with

LLMs, leveraging implicit world knowledge to ensure reliable, adaptive planning [169]. Together, these methods illustrate how structured frameworks and environmental understanding can transform LLMs into effective planners.

Chapter 3

Memory

Memory is fundamental to both human and artificial intelligence. For humans, it serves as the bedrock of cognition, a vast repository of experiences and knowledge that empowers us to learn, adapt, and navigate the complexities of the world. From infancy, our capacity to encode, store, and retrieve information underpins our ability to acquire language, master skills, and build relationships. Decades of research in neuroscience and cognitive psychology have illuminated the multifaceted role of memory, revealing its influence on our sense of self, creative endeavors, and decision-making processes. Similarly, in the burgeoning field of artificial intelligence, memory is increasingly recognized as a cornerstone of intelligent behavior. Just as humans rely on past experiences to inform present actions, AI agents require robust memory mechanisms to tackle intricate tasks, anticipate future events, and adjust to dynamic environments. Therefore, a deep understanding of human memory – its organization, processes, and limitations – provides invaluable insights for the development of more capable and adaptable AI systems. This section will first provide a concise overview of human memory, focusing on the key stages of encoding, consolidation, and retrieval. We will then transition to exploring the diverse approaches employed in designing AI agent memory systems, ranging from traditional symbolic representations to cutting-edge neural network-based methods. A critical comparison between these artificial memory systems and their human counterparts will highlight existing gaps in areas such as adaptability, contextual understanding, and resilience. Finally, we will consider how principles derived from neuroscience and cognitive psychology can inform future research, suggesting directions that may lead to the creation of artificial memory systems that exhibit greater robustness, nuance, and ultimately, a closer resemblance to the remarkable capabilities of human memory.

3.1 Overview of Human Memory

3.1.1 Types of Human Memory

Human memory is often conceptualized as a multi-tiered system that captures, stores, and retrieves information at different levels of processing and timescales. Researchers from the fields of cognitive science, neuroscience, and psychology have proposed various models to describe these levels. A commonly accepted hierarchy distinguishes between sensory memory, short-term memory (including working memory), and long-term memory [170, 171]. Within long-term memory, explicit (declarative) and implicit (non-declarative) forms are further delineated [172]. Figure 3.1 illustrates one such hierarchical framework:

- **Sensory Memory.** Sensory memory is the initial, brief store of raw sensory information. It maintains inputs from the environment for a duration ranging from milliseconds to a few seconds, allowing subsequent processes to determine which portions of the stimulus are relevant for further analysis [173]. Iconic memory (for visual input) [174] and echoic memory (for auditory input) [175] are two well-known subtypes.
- **Short-Term Memory and Working Memory.** Short-term memory (STM) involves holding a limited amount of information in an easily accessible state for seconds to under a minute. The term *working memory* is often used to emphasize the manipulation of that information rather than mere maintenance. While some models treat working memory as a subset of STM, others view it as a distinct system that manages both the storage and active processing of data (for instance, performing arithmetic in one’s head) [176, 177]. The capacity of STM or working memory is finite, typically cited as around seven plus or minus two chunks of information [98], though individual differences and task factors can modulate this figure.

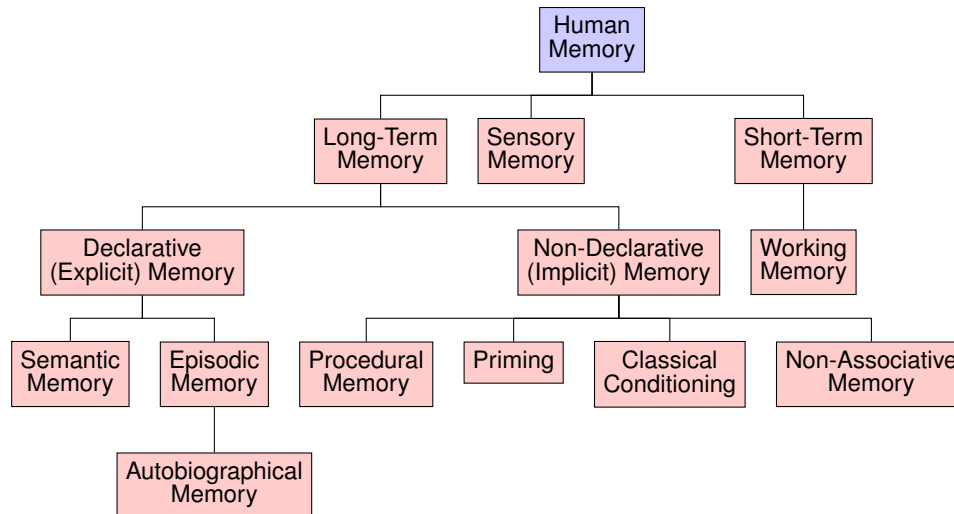


Figure 3.1: The hierarchical taxonomy of human memory system.

- **Long-Term Memory (LTM).** Long-term memory accommodates the more durable storage of information that can persist from hours to decades [178, 179]. This repository supports the learning of skills, the acquisition of factual knowledge, and the recollection of personal experiences. Although long-term memory is sometimes described as having a vast or near-unlimited capacity, factors such as decay, interference, and retrieval cues influence the extent to which stored information can be accessed [180].
 - **Declarative (Explicit) Memory.** Declarative memory encompasses memories that can be consciously recalled and articulated [181]. Within this broad category, researchers often discuss:
 - * **Semantic Memory:** Factual knowledge about the world, including concepts, words, and their relationships [182]. Examples include recalling the meaning of vocabulary terms or knowing the capital city of a country.
 - * **Episodic Memory:** Personally experienced events that retain contextual details such as time, place, and the people involved [183]. This form of memory allows individuals to mentally travel back in time to relive past experiences.
 - * **Autobiographical Memory:** A form of episodic memory focusing on events and experiences related to one's personal history [184]. While sometimes treated as a sub-category of episodic memory, autobiographical memory places particular emphasis on the self and its evolving life narrative.
 - **Non-Declarative (Implicit) Memory.** Non-declarative memory refers to memories that influence behavior without the need for conscious awareness [185]. Key subtypes include:
 - * **Procedural Memory:** The gradual acquisition of motor skills and habits (e.g., riding a bicycle, typing on a keyboard) that become automatic with repetition [186, 187].
 - * **Priming:** The phenomenon in which prior exposure to a stimulus influences subsequent responses, often without explicit recognition of the previous encounter [188].
 - * **Classical Conditioning:** The learned association between two stimuli, where one stimulus comes to elicit a response originally produced by the other [189].
 - * **Non-Associative Memory:** Adaptive modifications in behavior following repeated exposure to a single stimulus. Habituation (reduced response to a repeated, harmless stimulus) and sensitization (increased response after exposure to a noxious or intense stimulus) are representative examples [190, 191].

Despite the orderly appearance of these categories, human memory processes often overlap. For example, autobiographical memory is typically nested within episodic memory, yet its particular focus on self-relevant experiences leads some theorists to treat it as a slightly different category. Similarly, the boundary between short-term and working memory can differ depending on the theoretical perspective. Some scholars prefer a more functional, process-oriented view of working memory, while others employ a strictly capacity-oriented concept of short-term storage. In each case, these different perspectives on memory highlight the complexity and nuance of human cognition.

3.1.2 Models of Human Memory

Human memory has inspired a wide range of theoretical models, each offering different insights into how information is acquired, organized, and retrieved. Although no single framework commands universal agreement, several influential perspectives have shaped the discourse in cognitive science, neuropsychology, and AI research. The following content highlights some of the most prominent models and architectures used to explain memory’s multiple facets.

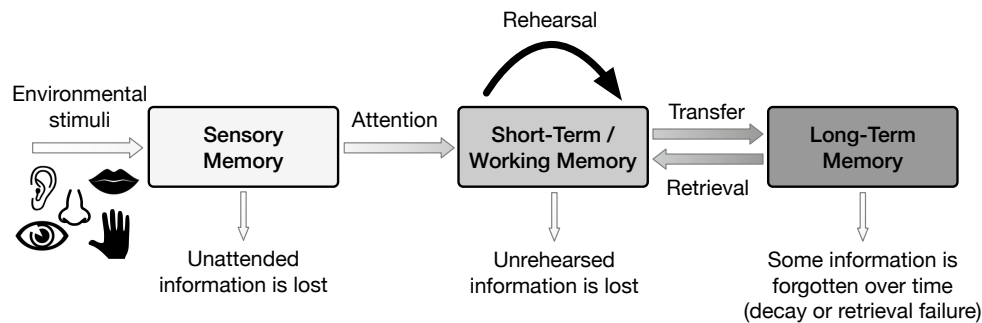


Figure 3.2: Atkinson-Shiffrin three-stage model of human memory [170].

The Multi-Store (Modal) Model. A seminal proposal by Atkinson and Shiffrin [170] introduced the multi-store or “modal” model, which posits three main stores for incoming information: *sensory memory*, *short-term memory*, and *long-term memory*. Control processes (e.g., attention, rehearsal) regulate how data transitions across these stores. Figure 3.2 illustrates this model of memory. Despite its relative simplicity, this model remains foundational for understanding how fleeting sensory impressions eventually form stable, long-lasting representations.

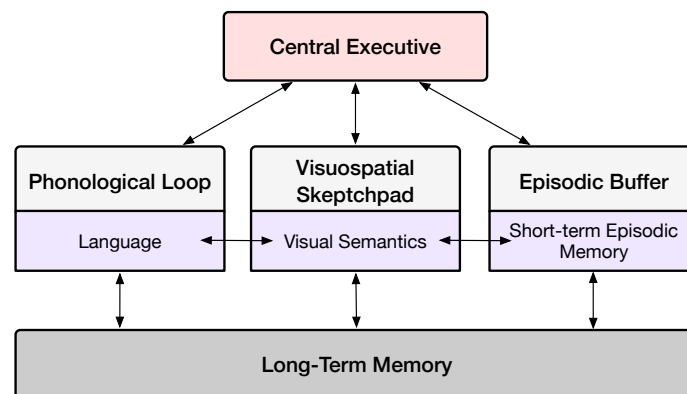


Figure 3.3: Baddeley’s model of working memory [192].

Working Memory Models. Recognizing that short-term memory also involves active maintenance, Baddeley and Hitch [192] proposed a *working memory* framework emphasizing the dynamic manipulation of information. Their original model described a central executive that coordinates two subsystems: the phonological loop (verbal) and the visuospatial sketchpad (visual/spatial). A subsequent refinement introduced the episodic buffer to integrate material from these subsystems with long-term memory [193]. Figure 3.3 shows the framework of the working memory model. Alternatives such as Cowan’s embedded-processes model [194] similarly underscore the role of attention in governing how information is briefly sustained and manipulated.

Serial-Parallel-Independent (SPI) Model. Initial distinctions between episodic, semantic, and procedural memory were championed by Tulving [195], who later refined his ideas into the Serial-Parallel-Independent (SPI) model, as shown in Figure 3.4. In this framework, memory is divided into two overarching systems. The *cognitive representation system* handles perceptual input and semantic processes, encompassing facts, concepts, and contextual (episodic) knowledge. The *action system*, by contrast, underpins procedural skills such as dance routines, driving maneuvers, or typing proficiency. Tulving’s SPI model posits that memory formation can occur at multiple levels: strictly perceptual encoding can support rudimentary episodic memories, while richer episodic representations benefit from semantic

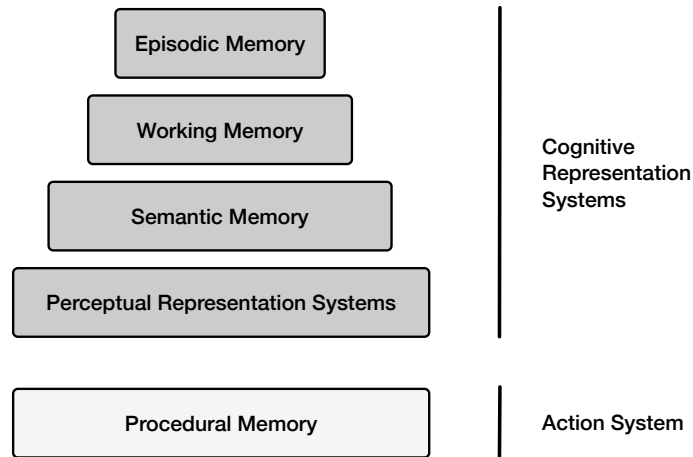


Figure 3.4: The Serial-Parallel Independent (SPI) model of human memory [195].

mediation. For instance, patients with semantic dementia, who struggle to retain word meanings, can still form some episodic memories but often lack the full contextual detail conferred by intact semantic networks. By highlighting the role of procedural memory and its automatic, intuitive nature, the SPI model aims to integrate structure (the content of memory) and function (how memory is used), surpassing earlier accounts that largely focused on explicit storage and retrieval. Despite these strengths, critics note that the model under-specifies how working memory operates within the broader system, and the feedback mechanisms connecting cognitive and action subsystems remain loosely defined.

Global Workspace Theory (GWT) and the IDA/LIDA Framework. Global Workspace Theory (GWT), developed by Baars [196], conceptualizes consciousness and working memory as a “broadcast” mechanism that distributes information to specialized processors. Building on GWT, Franklin [197, 198] proposed the *IDA (Intelligent Distribution Agent)* model, later extended to *LIDA (Learning IDA)*, as a comprehensive cognitive architecture. In these frameworks, multiple memory systems (e.g., perceptual, episodic, procedural) interact through iterative “cognitive cycles”, with a global workspace functioning as a hub for attention and decision-making. From an AI standpoint, IDA/LIDA demonstrates how human-like memory processes can be operationalized to guide an agent’s perception, action selection, and learning.

ACT-R and Cognitive Architectures. ACT-R (Adaptive Control of Thought—Rational) [199] is a comprehensive cognitive architecture that integrates memory, perception, and motor processes into a unified theoretical framework. It has been applied extensively across diverse domains, including learning and memory, problem-solving, decision-making, language comprehension, perception and attention, cognitive development, and individual differences. Figure 3.5 illustrates the processes of ACT-R. At the core of ACT-R are distinct *modules* (e.g., visual, manual, declarative, procedural) that interact with the system through dedicated *buffers*. Declarative memory stores factual “chunks,” while procedural memory encodes if–then production rules for actions and strategies. Cognition unfolds via a *pattern matcher* that selects a single production to fire based on the current buffer contents. This symbolic production system is augmented by subsymbolic processes, guided by mathematical equations that dynamically regulate activations, retrieval latencies, and production utilities. By combining symbolic and subsymbolic levels, ACT-R provides a mechanistic account of how individuals acquire, retrieve, and apply knowledge—thus shedding light on empirical phenomena such as reaction times, error patterns, and the shaping of learning over time.

Each of the aforementioned models illuminates different aspects of memory. The multi-store model provides a straightforward introduction to storage stages, working memory models emphasize active maintenance and manipulation, and frameworks such as IDA/LIDA or ACT-R embed memory within a comprehensive view of cognition. In practice, researchers often draw upon multiple perspectives, reflecting the intricate nature of human memory and its integral role in perception, learning, and adaptive behavior.

3.2 From Human Memory to Agent Memory

Having established the fundamentals of human memory, we now focus on how Large Language Model (LLM)-based agents manage and store information. Memory is not merely a storage mechanism but is fundamental to human and

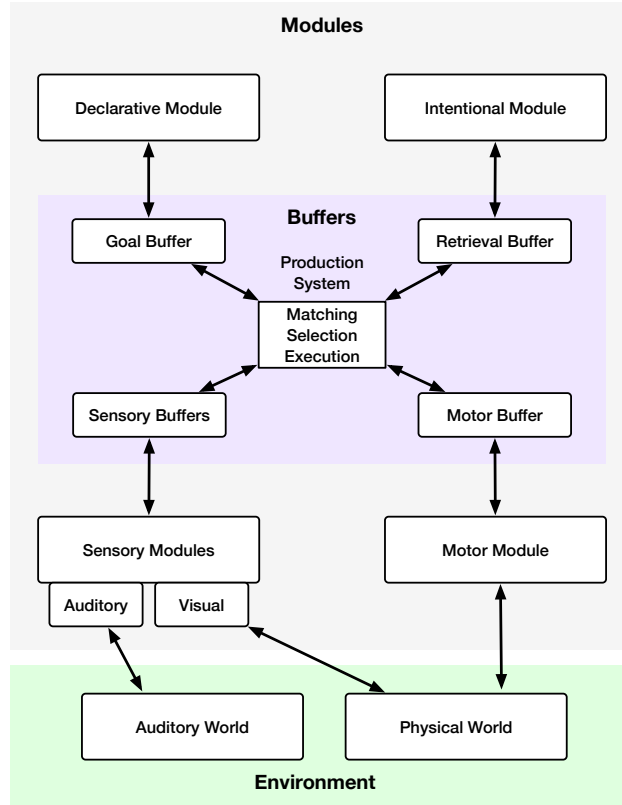


Figure 3.5: An abstraction of the most important processes in the ACT-R model [199].

artificial intelligence. Memory underpins cognition, enabling learning, adaptation, and complex problem-solving for humans. Similarly, for LLM-based agents, memory provides the crucial scaffolding for maintaining context, learning from experience, and acting coherently over time. Without memory, even a highly capable LLM would struggle to adapt to changing circumstances or maintain focus during extended interactions.

While LLM-based agents and biological systems differ fundamentally, the principles guiding human memory—context retention, selective forgetting, and structured retrieval—are highly relevant to agent design. Therefore, examining the parallels and distinctions between human and artificial memory is beneficial. Functionally, we can draw analogies: an agent’s short-term memory buffer resembles the prefrontal cortex’s role in working memory, while long-term storage in a vector database is akin to the hippocampus’s function in consolidating episodic memories. Agent memory design can benefit from emulating human memory’s mechanisms, including selective attention, prioritized encoding, and cue-dependent retrieval. However, crucial differences exist.

Human memory, built upon biological neural networks, integrates storage and computation within neurons’ connections and activity patterns. This offers a high degree of parallelism and adaptability. In contrast, current agent memory systems predominantly rely on digital storage and algorithms, using symbolic representations and logical operations, thus separating storage and computation. This impacts information processing: human memory is associative and dynamic, capable of fuzzy matching and creative leaps, while current agent memory relies on precise matching and vector similarity, struggling with ambiguity. Although digital storage capacity is vast, it cannot yet replicate the complexity and dynamism of human memory, particularly in nuanced pattern recognition and long-term stability. Human memory, while imperfect, excels at extracting crucial information from noisy data. Agent memory systems, in their current stage, are still nascent compared to the intricacies of human memory, facing limitations in organization, integration, adaptive forgetting, and knowledge transfer.

The need for a dedicated memory module in LLM-based agents is paramount. While external knowledge bases (databases, search engines, APIs) [200] provide valuable information, they do not capture the agent’s internal reasoning, partial inferences, or task-specific context. An agentic memory system internalizes interim steps, evolving objectives, and historical dialogue, enabling self-referential exploration and adaptation. This is crucial for tasks requiring the agent to build upon prior judgments or maintain a personalized understanding of user goals.

Early approaches to agent memory, such as appending conversation history to the input prompt (a rudimentary form of working memory) [201], have evolved. Modern architectures employ more sophisticated techniques, including vector embeddings for rapidly retrieving memories [202] and selective incorporation of reasoning chains into subsequent inference steps [203, 204]. These diverse methods share the common goal of managing a large information reservoir without compromising system responsiveness.

However, compared to the sophistication of human memory, current agentic methods have limitations. Many systems lack coherent strategies for long-term memory consolidation, leading to cluttered logs or abrupt information loss. The flexible, bidirectional interplay between stored knowledge and ongoing processing, characteristic of human working memory, is often absent. Metacognitive oversight—selective recall, forgetting, and vigilance against outdated information—is also underdeveloped in LLM-based agents. Balancing comprehensive recall with practical efficiency, as humans do, remains a key challenge.

Building robust and adaptable memory for LLM-based agents involves addressing three core research questions: First, how should memory be represented to capture diverse information types and facilitate efficient access? Second, how can agent memory evolve, incorporating new experiences, adapting to changing contexts, and maintaining consistency? Finally, how can the stored memories effectively enhance reasoning, decision-making, and overall agent performance? The following sections delve into these crucial areas, exploring current approaches, limitations, and potential future directions.

3.3 Representation of Agent Memory

Inspired by human cognitive systems [285], current memory architecture in intelligent agents adopts a hierarchical framework that integrates perception through sensory memory [205], real-time decision-making via short-term memory [286, 287], and sustained knowledge retention through long-term memory [288, 289, 48]. This multi-layered structure equips agents to manage immediate tasks while maintaining a broader contextual understanding, fostering adaptability and seamless continuity across diverse interactions.

Specifically, the memory system transforms raw environmental inputs into structured, actionable representations. Sensory memory acts as the gateway, capturing and selectively filtering perceptual signals to provide a foundation for cognitive processing. Short-term memory bridges these immediate perceptions with task-level understanding, buffering recent interactions and enabling dynamic adaptation through experience replay and state management. Long-term memory then consolidates and stores information over extended periods, facilitating cross-task generalization and the accumulation of enduring knowledge.

Together, these memory components form a cohesive cycle of perception, interpretation, and response. This cycle supports real-time decision-making and enables agents to learn and evolve continuously, reflecting an intricate balance between responsiveness and growth. The following delves into the formulation of each memory type, exploring their unique roles and interactions within the agent’s cognitive architecture.

3.3.1 Sensory Memory

In human cognitive systems, sensory memory serves as a mechanism for collecting information through the senses—touch, hearing, vision, and others—and is characterized by its extremely brief lifespan. Analogously, sensory memory functions as the embedded representation of inputs such as text, images, and other perceptual data in intelligent agents. It represents the initial phase of environmental information processing, acting as a gateway for transforming raw observations into meaningful representations for further cognitive processing.

Sensory memory in intelligent agents transcends passive information reception. It dynamically encodes and filters perceptual signals, bridging immediate sensory inputs with the agent’s internal state, objectives, and prior knowledge. This adaptive process facilitates rapid perception of environmental changes, task continuity, and real-time context-aware information processing. Sophisticated attention mechanisms are employed to ensure relevance and focus in the sensory memory layer, forming a critical foundation for decision-making and adaptation.

Formally, sensory memory formation consists of three sequential steps: *perceptual encoding*, *attentional selection*, and *transient retention*. First, perceptual encoding transforms raw sensory signals into processable representations, mathematically expressed as:

$$\phi(o_t) = \text{Encode}(o_t, s_t) \quad (3.1)$$

where o_t is the sensory input at time t , and s_t represents the agent’s state. For instance, RecAgent [205] employs an LLM-based sensory memory module to encode raw observations while filtering noise and irrelevant content. Extending

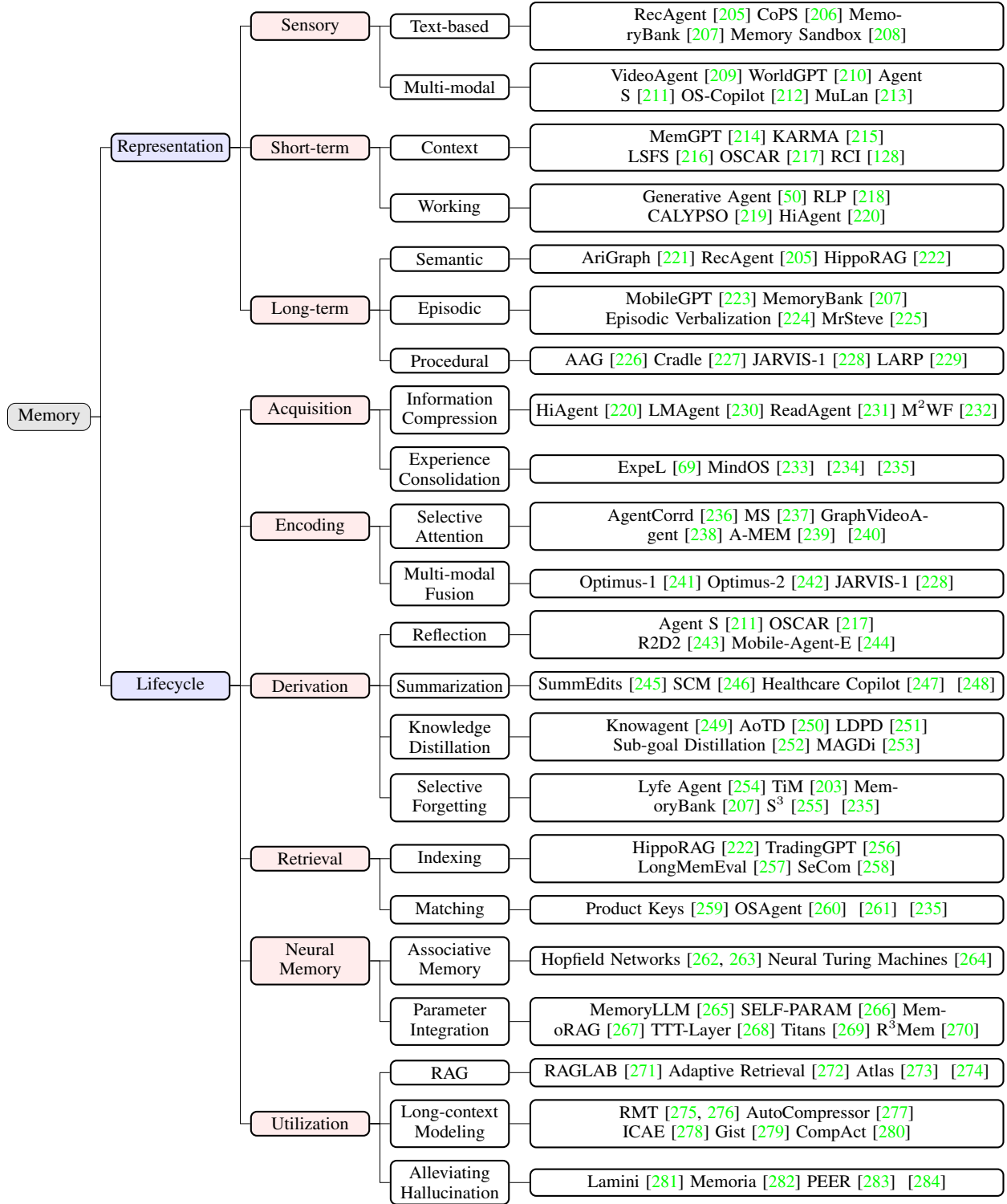


Figure 3.6: Tree diagram of the memory module in intelligent agents.

beyond text-based perception, multimodal sensory memory systems such as Jarvis-1 [228], VideoAgent [209], and WorldGPT [210] integrate multimodal foundation models to process diverse modality inputs.

Next, attentional selection extracts crucial information from the encoded sensory data. This process, guided by an attention mechanism, is represented as:

$$\alpha_t = \text{Attention}(\phi(o_t), c_t) \quad (3.2)$$

where $\phi(o_t)$ is the encoded input, and c_t denotes contextual information influencing attention. For example, RecAgent [205] employs an attention mechanism with an importance scoring system that assigns relevance scores to compressed observations, prioritizing critical inputs such as item-specific interactions while de-emphasizing less significant actions. This helps extract high-priority information for memory retention.

Finally, transient retention temporarily stores the selected sensory information as sensory memory:

$$M_{\text{sensory}} = \{\alpha_t \mid t \in [t - \tau, t]\} \quad (3.3)$$

Several strategies have been implemented to manage the time window. For instance, RecAgent [205] models retention by associating each observation with the timestamp corresponding to the start of a simulation round in the user behavior simulation environment, represented as a triplet (observation, importance score, timestamp). Similarly, CoPS [206] employs a fixed-size sensory memory pool as a time window, which consists of user search requests for personalized search, facilitating “re-finding” behavior. When a new query is received, the system first checks the sensory memory for relevant matches. If a match is found, the query is classified as a re-finding instance, enabling a rapid sensory response.

3.3.2 Short-Term Memory

Short-term memory in cognition-inspired intelligent agents serves as a transient and dynamic workspace that bridges sensory memory and long-term memory. It is essential for storing and processing task-relevant information and recent interaction sequences, supporting real-time decision-making and adaptive behavior. Inspired by human short-term and working memory, it temporarily retains information to facilitate complex cognitive tasks, ensuring continuity and coherence in the agent’s operations.

Short-term memory in intelligent agents can be categorized into *context memory* and *working memory*. On the one hand, context memory treats the context window as the short-term memory of LLMs. For example, MemGPT [214], inspired by hierarchical memory systems in operating systems, manages different storage tiers to extend context beyond the LLM’s inherent limitations. [290] introduces a neurosymbolic context memory that enhances LLMs by enabling symbolic rule grounding and LLM-based rule application.

On the other hand, working memory involves fetching and integrating relevant external knowledge to hold essential information during an agent’s operation. Generative Agent [50] employs short-term memory to retain situational context, facilitating context-sensitive decision-making. Reflexion [48] utilizes a sliding window mechanism to capture and summarize recent feedback, balancing detailed immediate experiences with high-level abstractions for enhanced adaptability. RLP [218] maintains conversational states for speakers and listeners, using them as short-term memory prompts to support dialogue understanding and generation.

For interactive and creative game scenarios, CALYPSO [219] assists Dungeon Masters in storytelling for Dungeons & Dragons by constructing short-term memory from scene descriptions, monster details, and narrative summaries, enabling adaptive storytelling and dynamic engagement. Similarly, Agent S [211] and Synapse [291], designed for GUI-based autonomous computer interaction, define their short-term memory as task trajectories, including actions such as button clicks and text inputs. This formulation supports behavioral cloning and enhances adaptation in novel GUI navigation tasks.

In robotics applications, SayPlan [292] leverages scene graphs and environmental feedback as short-term memory to guide planning and execution in scalable robotic environments. KARMA [215] engages short-term working memory with an effective and adaptive memory replacement mechanism to dynamically record changes in objects’ positions and states. LLM-Planner [293] iteratively updates short-term memory with environmental observation to prompt an LLM for dynamic planning.

3.3.3 Long-Term Memory

Long-term memory in cognition-inspired intelligent agents enables the retention and retrieval of information over extended periods, allowing agents to generalize knowledge and adapt to new contexts effectively. Unlike sensory and short-term memory, which handle transient or immediate data, long-term memory supports cumulative learning and cross-task adaptability. It mirrors human long-term memory by incorporating explicit and implicit components, facilitating richer contextual understanding and intuitive behavior.

On the one hand, *explicit memory* involves intentional recollection, analogous to declarative memory in humans. It consists of *semantic memory*, which stores general knowledge such as facts and concepts, and *episodic memory*,

which records specific events and interaction histories. Semantic memory in intelligent agents can be preloaded from domain knowledge bases or dynamically acquired through interactions. For example, in environments like TextWorld, semantic memory captures structured facts, such as “*Recipe – contains – Tuna*” or “*Recipe – is on – Table*”. Episodic memory, in contrast, logs situational context and sequential actions, such as “go from kitchen to living room, then to garden”. Integrating semantic and episodic memory allows agents to retain static and contextual information, enabling human-like adaptability and context-aware responses.

On the other hand, *implicit memory* shapes agent behavior through *procedural memory* and *priming*. Procedural memory enables agents to perform repetitive tasks efficiently by recalling specific skills and reusable plans. For example, it automates routine tasks without requiring explicit instructions, improving task execution efficiency. Priming, meanwhile, captures state changes and corresponding responses, allowing agents to adapt to similar contexts quickly. Priming enhances fluidity and context-sensitive decision-making by directly matching observations to or continuously chaining actions. Implicit memory, shaped by interactions with cognitive modules, enables rapid adaptation, often after minimal exposure to new stimuli.

Most intelligent agents implement both semantic and episodic memory within their memory modules. For instance, Agent S [211], designed for GUI automation tasks, incorporates semantic memory to store online web knowledge in natural language form, while episodic memory captures high-level, step-by-step task experiences. Similarly, AriGraph [221], targeting embodied simulation tasks, encodes semantic environment knowledge using a fact graph and logs episodic navigation history through an event graph. In AI companion systems like MemoryBank [207] for SiliconFriend, semantic memory constructs user portraits in natural language, while episodic memory retains interaction histories, enhancing personalized and context-aware behavior.

For implementing implicit memory, current agent systems primarily adopt model-friendly memory formats, such as key-value pair storage, executable code, or reusable routines. For example, AAG [226] defines and generalizes procedures through analogy, mapping knowledge from one situation (base) to another (target). This structure can be represented as a linear directed chain graph, where the input serves as the root, the output as the leaf node, and each intermediate step as a node in the chain. Similarly, Cradle [227] and Jarvis-1 [228] implement procedural memory by storing and retrieving skills in code form, which can be either learned from scratch or pre-defined. Once curated, skills can be added, updated, or composed within memory. The most relevant skills for a given task and context are then retrieved to support action planning.

3.4 The Memory Lifecycle

In this section, we introduce the lifecycle of memory in AI agents, as depicted in Figure 3.7. The lifecycle comprises a dual process of retention and retrieval. Retention includes acquisition, encoding, and derivation, while retrieval involves memory matching, neural memory networks, and memory utilization.

3.4.1 Memory Acquisition

Memory Acquisition is the foundational process by which intelligent agents take in raw perceptual information from their environment. This initial step is crucial for subsequent learning, adaptation, and decision-making [305]. A primary challenge in acquisition is the sheer volume and complexity of environmental inputs. Agents are constantly bombarded with visual, auditory, textual, and other forms of data, much of which is redundant or irrelevant to the agent’s goals. Therefore, a core aspect of memory acquisition is not simply capturing data, but also initiating a preliminary filtering process. This filtering leverages two primary mechanisms: initial *information compression* and *experience consolidation*.

At this early stage, information compression involves rudimentary techniques to reduce data dimensionality. This might include downsampling images, extracting key phrases from text using simple heuristics, or identifying significant changes in audio streams [306]. The goal is rapid, lossy compression to prioritize potentially relevant information. For example, LMAgent [230] prompts the LLM to perform information compression, reducing irrelevant and unimportant content when constructing sensory memory to enhance operational efficiency. Meanwhile, ReadAgent [231] and GraphRead [307] respectively employ different strategies for compressing long text, i.e., episode pagination and graph-based structuring, to maximize information retention while ensuring efficiency.

On the other hand, experience consolidation, even at the acquisition phase, plays a role. The agent doesn’t yet have a rich memory, but it can begin to apply previously learned, very general rules or biases. For example, if the agent has a pre-existing bias towards moving objects, it might prioritize visual data containing motion, even before full encoding [308]. To enhance the dynamic consolidation of memory-based experiences, [235] define metrics such as contextual relevance and recall frequency to determine whether to update long-term memory in a vector database.

Method	Domain	Memory Representation			Memory Lifecycle				
		Sensory	Short-term	Long-term	Acquisition	Encoding	Derivation	Retrieval	Utilization
Synapse [291]	GUI	Multi-modal	Context	Episodic, Procedural	User demo.	-	Hierarch. Decomp.	-	-
Agent S [211]	GUI	Multi-modal	Context, Working	Semantic, Episodic	Info. Compress.	Contrastive Learn.	Select. Forget.	Indexing	Long-context
Automanual [108]	GUI	Multi-modal	Context	Procedural, Episodic	User Demo.	Hierarch. Parse	Goal Decomp.	Task Search	Subgoal Exec.
AutoGuide [294]	GUI	Multi-modal	Context	-	Screen Capture	-	Action Plan	-	Action Exec.
Agent-Pro [295]	GUI	Multi-modal	Context	-	Screen Capture	-	Hierarch. Decomp.	-	Action Exec.
MemGPT [214]	Document	Text	Context, Working	-	External Data	-	-	Paging, Func. call	Doc. interact.
SeeAct [296]	Web	Multi-modal	Context	-	Screen Capture	-	Action Plan	-	Web Interact.
AutoWebGLM [297]	Web	Text	Context	-	HTML Parse	HTML Embed	HTML Analysis	-	Web Interact.
SteP [298]	Web	Text	Context	Task-spec.	HTML Parse	HTML Embed	HTML Analysis	Element Rank	Web Interact.
AWM [299]	Web	Text	-	Procedural	Workflow Extract.	Action Summ.	-	Sim. lookup	Workflow exec.
AriGraph [221]	TextWorld	Text	-	Semantic, Episodic	Env. Observ.	Knowl. Graph	Graph Traversal	Assoc. Retrieval	Action plan.
MemoryBank [207]	Dialogue	Text	-	Episodic	Dialogue Record	-	-	Chron. order	Resp. gen.
PromptAgent [300]	General	Text	Context	-	Prompting	-	Prompt Refine.	Content-based	Prompt Exec.
ECL [301]	Embodiment	Multi-modal	Context	Episodic	Obs. Recording	Contrast. Learn.	Exper. Summ.	Sim. & Recency	Policy Learn.
LEO [302]	Embodiment	Multi-modal	Working	Long-Horizon Rep.	Observation	Spatial-Temp. Learn.	Goal-Cond. Policy	Hierarch. Plan	Long-Horizon Exec.
IER [303]	Embodiment	Multi-modal	Context	Episodic	Env. Interact.	Multi-modal Embed	Iter. Refine.	Sim. Match	Action Plan.
Voyager [47]	Embodiment	Text	Working	Procedural	Auto. Curriculum	Skill Library	Iter. Prompt.	-	Skill Exec.
A3T [49]	Embodiment, Robotics	Text	Context	-	Task Decomp.	Token. & Embed.	Action Planning	-	Action select.
STARLING [304]	Robotics	Multi-modal	Context	Procedural	Demo.	Traj. Encode	Skill Refine.	Sim. & Context	Skill Exec.

Table 3.1: Summary of the memory module in various agents. Refer to Figure 3.6 for abbreviations.

Expel [69] constructs an experience pool to collect and extract insights from training tasks, facilitating generalization to unseen tasks. More recently, MindOS [233] proposed a working memory-centric central processing module for building autonomous AI agents, where working memory consolidates task-relevant experiences into structured thoughts for guiding future decisions and actions.

These two mechanisms work in concert with preliminary LLM input. To address the initial challenges, several mechanisms have to be deployed. Agents must be equipped with mechanisms to assess the potential relevance of incoming information rapidly. This preliminary filtering prevents cognitive overload. The acquisition phase also benefits from LLM.

3.4.2 Memory Encoding

Memory encoding builds upon acquisition by transforming the filtered perceptual information into internal representations suitable for storage and later use. A key aspect of encoding is selective filtering. This selective attention mimics human cognitive processes [309]. The inherent challenges of encoding stem from the complexity, high dimensionality, and often noisy nature of raw perceptual data. Effective encoding requires advanced mechanisms to identify key

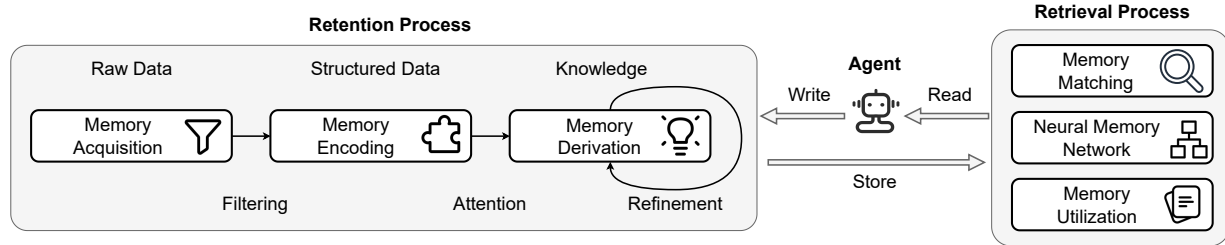


Figure 3.7: Illustration of the memory lifecycle. The memory retention process involves three sequential steps—memory acquisition, encoding, and derivation, while the memory retrieval process encompasses several independent applications, including matching (vector search), neural memory networks, and memory utilization (for long-context modeling and hallucination mitigation).

features, compress them compactly, and integrate information from multiple modalities. Modern approaches address these challenges by leveraging *selective attention* and *multi-modal fusion*.

Selective Attention mechanisms, inspired by human cognition, allow the agent to dynamically focus computational resources on the most relevant parts of the input. This might involve attending to specific regions of an image, keywords in a text, or particular frequencies in an audio signal. Different attention mechanisms can be used depending on the modality and task. For example, as the candidate memory dynamically expands, MS [237] employs an LLM-based scorer to selectively retain the top-scoring half, creating a more compact shared memory across multiple agent systems. In other modalities, GraphVideoAgent [238] utilizes graph-based memory to enable selective and multi-turn video scene understanding, enhancing question-answering performance. In robot control, [240] implements selective attention as a filtering mechanism to extract task-relevant objects from the set of all perceived objects on the table.

Multi-modal Fusion [310] is essential for integrating information from different sensory inputs (e.g., combining visual and auditory data to understand a scene). This involves creating a unified representation space where features from different modalities are aligned. Cross-modal encoders and contrastive learning techniques are often used to achieve this fusion. For example, JARVIS-1 [228] uses the general-domain video-language model CLIP [51] to compute alignment within a multimodal key-value memory, where the key comprises elements such as task, plan, and visual observations, and the value is a text-based representation of successfully executed plans. Furthermore, Optimus-1 [241] refines memory representation and optimizes the multimodal encoder by leveraging MineCLIP [311], a domain-specific video-language model pre-trained on Minecraft gameplay, to align and fuse filtered video streams with textual instructions and plans, encoding the agent’s multimodal experiences into an abstracted memory pool. This integrated representation enhances information retrieval and reasoning across modalities and acts as another filter, reinforcing consistent data. LLMs’ semantic understanding is utilized to extract relevant features efficiently.

3.4.3 Memory Derivation

Memory derivation focuses on extracting meaningful knowledge and insights from the acquired and encoded memories. This process goes beyond simple storage. This stage is essential for enhancing the agent’s learning capabilities. The goal is to continuously optimize the structure and content of the agent’s memory. A significant challenge in derivation is the dynamic evaluation of information value. Strategies to address these challenges include *reflection*, *summarization*, *knowledge distillation*, and *selective forgetting*.

Reflection involves an agent actively analyzing its memories to identify patterns, relationships, and potential inconsistencies. It can be triggered by specific events (e.g., an unexpected outcome) or occur periodically as a background process. This process may include comparing memories, reasoning about causal relationships, and generating hypotheses [300]. ExpeL [69] leverages reflection to collect past experiences for generalization to unseen tasks and to support trial-and-error reattempts following failures. R2D2 [243] models memory as a replay buffer and applies reflection to refine it by correcting failed execution trajectories in web agents. These corrected trajectories are then combined with successful ones to construct reflective memory, which serves as a reference for future decision-making.

Summarization aims to produce concise representations of larger bodies of information while preserving their most essential content. This can include extracting key sentences from a document, generating abstractive summaries of conversations, or condensing sequences of events. Summarization techniques range from simple extractive methods to advanced abstractive approaches powered by large language models (LLMs) [245, 312, 246]. For example, [248] introduces a recursive summarization strategy over dialogue history and prior memory to support long-term dialogue memory derivation. Building on this, Healthcare Copilot [247] maintains concise memory by transforming conversation

memory, representing the full ongoing medical consultation, into history memory that retains only key information relevant to the patient’s medical history.

Knowledge distillation [313] enables agents to transfer knowledge from larger, more complex models (or ensembles) to smaller, more efficient ones. This is particularly important for resource-constrained agents and for enhancing generalization. Distillation can also involve consolidating knowledge from multiple specialized models into a single, general-purpose model. For example, AoTD [250] distills textual chains of thought from execution traces of subtasks into a Video-LLM to enhance multi-step reasoning performance in video question answering tasks. LDPD [251] transfers decision-making outcomes from teacher agents (i.e., expert buffers) to student agents, optimizing the student’s policy to align with the teacher’s. In multi-agent systems, MAGDi [253] distills the reasoning interactions among multiple LLMs into smaller models by structurally representing multi-round interactions as graphs, thereby improving the reasoning capabilities of smaller LLMs.

Selective forgetting [314] is the crucial process of removing or down-weighting memories that are deemed irrelevant, redundant, or outdated. This is essential for maintaining memory efficiency and preventing cognitive overload. Forgetting mechanisms can be based on time (older memories are more likely to be forgotten) [247], usage frequency (infrequently accessed memories are more likely forgotten) [203], and relevance to the current task or context [255]. In more fine-grained forgetting mechanisms, MemoryBank [207] applies the Ebbinghaus Forgetting Curve to quantify the forgetting rate, accounting for both time decay and the spacing effect, i.e., the principle that relearning information is easier than learning it for the first time. In contrast, Lyfe Agent [254] adopts a hierarchical summarize-and-forget strategy: it first clusters related memories, refines them into concise summaries, and then removes older memories that are highly similar to newer ones. This approach enables efficient, low-cost memory updates for real-time social interactions.

3.4.4 Memory Retrieval and Matching

Memory retrieval is a process that emulates the human ability to recall relevant knowledge and experiences to solve problems. The goal is to efficiently and accurately extract the most pertinent memory fragments from a large and diverse memory pool, encompassing sensory, short-term, and long-term memory, to inform the agent’s decisions, planning, and actions. Just as humans rely on past experiences to navigate complex situations, agents require a sophisticated memory retrieval mechanism to handle a wide range of tasks effectively.

However, achieving this goal presents several significant challenges. First, the agent’s memory repository is often heterogeneous, comprising various forms of memory such as natural language descriptions, structured knowledge graphs, and state-action-reward sequences. These memories differ fundamentally in their data structures, representations, and levels of semantic granularity, posing a challenge for unified retrieval. Second, the retrieved memory fragments must be highly relevant to the current context, including the agent’s state, task goals, and environmental observations. Simple keyword matching falls short of capturing the deeper semantic relationships required for meaningful retrieval. Developing a context-aware semantic matching mechanism that can dynamically adjust the retrieval strategy based on the current situation is therefore paramount. Third, the real-time nature of agent interaction with the environment necessitates efficient memory retrieval to support rapid decision-making and action [315]. This demand for efficiency is further compounded by the limitations of the agent’s computational resources. Finally, the agent’s memory is not static but constantly evolving as new experiences, knowledge, and skills are acquired. Ensuring memories’ timeliness, reliability, and relevance while avoiding the interference of outdated or erroneous information is a continuous challenge.

A comprehensive approach can address these challenges, encompassing four key components. Firstly, a foundational step involves constructing a unified memory representation and indexing scheme. This aims to bridge the representational gap between different memory types by embedding them into a common vector space. Pre-trained language models like BERT or Sentence-BERT [316] can be leveraged to transform text-based memories into semantic vectors, while graph neural networks (GNNs) can learn vector representations for structured memories like knowledge graphs, capturing both node and edge relationships [317]. To facilitate efficient retrieval, a multi-layered hybrid indexing structure is essential. This integrates techniques like inverted indexes for keyword matching, vector indexes like Faiss [318] or Annoy [319] for similarity search, and graph indexes for structural queries [320], thus supporting diverse query needs.

Secondly, perhaps most critically, the system must develop context-aware semantic similarity computation. This allows the retrieval process to understand and utilize the current context, such as the agent’s state, goals, and observations, enabling a deeper semantic match beyond keyword overlap. This involves encoding the contextual information into vector representations and effectively fusing them with memory vectors. The attention mechanism plays a crucial role here, dynamically calculating the relevance between context and memory vectors and assigning different weights to memory fragments based on their contextual relevance [261]. This emphasizes memories that are more pertinent to the current situation.

Thirdly, integrating memory retrieval with the agent’s task execution necessitates a task-oriented sequence decision and dynamic routing mechanism. This leverages the structural information of tasks to guide memory retrieval and utilization, enabling complex task decomposition, planning, and dynamic adjustments. By constructing a task dependency graph, the agent can topologically sort subtasks to determine execution order. During execution, each subtask’s goal serves as context for memory retrieval, extracting relevant knowledge and experience. Moreover, the agent must adapt to environmental feedback and task progress, dynamically adjusting the execution plan. Each decision point involves re-retrieving memories based on the current state and goal to select the optimal action and handle unexpected situations. This aspect also emphasizes how agents can leverage their skill memory to solve problems, including skill distillation, combination, and innovation. Pattern recognition allows for summarising general problem-solving steps, while structured knowledge organization arranges skills into a retrievable format. Agents can further distill generalized skills from specific ones, combine multiple skills to address complex challenges, and even innovate new skill combinations. These processes depend fundamentally on an efficient memory retrieval system that can identify appropriate skills or skill combinations based on task requirements.

Finally, a robust memory management mechanism is crucial for maintaining the memory pool’s timeliness, relevance, and efficiency. This mechanism should incorporate a forgetting and updating strategy, mirroring human forgetting mechanisms [321]. This might involve regularly purging outdated, redundant, or infrequently used memories based on time-based decay (weakening memory strength over time) and frequency-based decay (purging low-frequency memories). Simultaneously, when a memory fragment relevant to the current task is retrieved, its timestamp and access frequency are updated, increasing its importance and ensuring dynamic memory updates. Through these concerted efforts, LLM Agents can be equipped with a powerful, flexible, and context-aware memory retrieval and matching system, enabling them to effectively utilize their accumulated knowledge, support complex decision-making, and exhibit more intelligent behavior.

3.4.5 Neural Memory Networks

Neural Memory Networks represent a fascinating frontier in AI research. They aim to integrate memory seamlessly into the fabric of neural networks. This approach departs from traditional memory architectures by encoding memories directly within the network’s weights or activations, transforming the network into a dynamic, read-write memory storage medium. This tight integration promises significant advancements in efficiency and the utilization of stored information. However, realizing this vision presents several formidable challenges.

A primary concern is balancing memory capacity with stability. Encoding a vast amount of information within the finite parameters of a neural network while maintaining long-term stability poses a major hurdle. The network must be able to store a multitude of memories without succumbing to catastrophic forgetting or confusion between similar memories. Equally crucial is the development of effective mechanisms for memory read-write operations. The network needs to reliably write new information, update existing memories, and accurately retrieve stored information on demand, all while maintaining computational efficiency. Beyond simply storing memories, the ultimate goal is to endow neural networks with the ability to generalize from and reason with the information they store. This would empower them to perform higher-order cognitive functions beyond rote memorization, allowing for insightful connections and inferences based on past experiences. Several approaches are being explored to address these challenges, notably through *associative memory* and *parameter integration*.

On the one hand, associative memory, inspired by the interconnectedness of neurons in the brain, offers a promising avenue. Models like Hopfield networks [262, 263], leveraging energy functions, and Bidirectional Associative Memories (BAMs) [322], supporting hetero-associative recall, provide mechanisms for encoding and retrieving patterns based on the weights between neurons. Besides, Neural Turing Machines (NTMs) [264] and Memory-Augmented Neural Network (MANNs) [323, 324, 275, 265] augment neural networks with external memory modules, employing attention and summary mechanisms to interact with these memories.

On the other hand, parameter integration represents another key research direction, aiming to encode memory directly within a network’s weights. This facilitates the seamless integration of world knowledge and accumulated experience into the operational behavior of intelligent AI agents. For example, some prior works modify model parameters to enable continual learning by updating [325, 326, 327] or forgetting specific knowledge [328]. Other studies treat LLMs as standalone memory modules, incorporating world knowledge into their parameters during pre-training [329], post-training [330], and online deployment [331]. For instance, MemoryLLM [265] introduces memory tokens, while SELF-PARAM [266] leverages knowledge distillation to embed world knowledge and past AI agent experiences into model parameters. This approach is further augmented in the M+ model [332] with a long-term memory mechanism and a co-trained retriever, enhancing its ability to generalize to longer history memorization. Additionally, [333] employs encoded memory to facilitate further reasoning, thereby improving the generalization of stored knowledge. More recently, MemoRAG [267] and R³Mem [270] have been proposed to not only encode memory but also enable

reliable retrieval from neural memory networks, unifying the dual processes of memory storage and retrieval within a single model. This advancement contributes to the development of next-generation generative-based retrieval systems, which support lifelong AI applications. Furthermore, Titans [269] have been introduced to memorize test-time data points through meta-learning, enabling more efficient test-time cross-task generalization.

Future research will continue to focus on creating larger capacity and more stable neural memory models. Concurrently, developing more efficient and flexible memory read-write mechanisms will be crucial. A critical area of investigation will involve applying these memory-augmented networks to complex cognitive tasks, pushing the boundaries of what AI can achieve. Progress in this domain will unlock new possibilities for building intelligent agents that can learn, remember, and reason in a manner that is increasingly reminiscent of human cognition.

3.4.6 Memory Utilization

A critical aspect of agent design lies in memory utilization, which focuses on maximizing the value of stored memory segments for the current task. The core objective is to apply these memories effectively and appropriately to enhance reasoning, decision-making, planning, and action generation, ultimately boosting the agent’s performance and efficiency while avoiding the pitfalls of irrelevant or incorrect memory interference. Achieving this, however, presents several challenges.

One primary challenge is balancing the vastness of the memory store with its effective utilization. Agents must navigate a potential information overload, ensuring that relevant memories are fully leveraged without overwhelming the system. Another hurdle is the need for abstraction and generalization. Agents need to distill specific memory segments into more general knowledge and apply this knowledge to new and varied situations. Furthermore, the issue of hallucinations and incorrect memories within the LLM requires careful consideration. Preventing the generation of content that contradicts or misrepresents stored information is crucial, as is the ability to identify and rectify erroneous information that may reside within the memory store itself.

To address these challenges, several strategies are employed. *Retrieval-augmented generation (RAG)* [334] combines retrieval and generation models to enhance the LLM’s capabilities by drawing upon external knowledge sources. Unlike the methods mentioned in memory retrieval and matching, RAG focuses on integrating retrieved information into the generation process itself. When prompted, the agent retrieves relevant memory segments and incorporates them into the context provided by the generation model. This contextual enrichment guides the model towards more factual and informative outputs. For instance, when responding to a user’s query, the agent can first retrieve related entries from its knowledge base and then generate an answer based on this retrieved information, thus grounding the response in established knowledge. More recently, some studies have integrated memory modules with RAG, incorporating self-reflection [274] and adaptive retrieval mechanisms [272] to enhance both generation reliability and efficiency. For example, Atlas [273] leverages causal mediation analysis, while [284] employs consistency-based hallucination detection to determine whether the model already possesses the necessary knowledge—allowing for direct generation—or whether retrieval is required, in which case the model first retrieves relevant information before generating a response. In a unified framework, RAGLAB [271] offers a comprehensive ecosystem for evaluating and analyzing mainstream RAG algorithms. HippoRAG [222] employs a strategy inspired by the hippocampal indexing theory of human memory to create a KG-based index for memory and use Personalized PageRank for memory retrieval.

Furthermore, *long-context modeling* plays a vital role in managing extensive memory stores. This approach enhances the LLM’s ability to process long sequences and large-scale memories, allowing for a deeper understanding and utilization of long-range dependencies. By employing Transformer model variants like Transformer-XL [324] and Longformer [335], or through hierarchical and recursive processing techniques, such as recurrent memory transformer (RMT) [275, 276], agents can expand their context window. This enables them to handle significantly more extensive memory stores and reason and make decisions within a much broader context. For example, agents can maintain a longer memory span when processing extensive documents or engaging in prolonged conversations. Additionally, some studies leverage memory to compress long contexts, enabling more effective long-context modeling. For example, AutoCompressor [277] introduces summary vectors as memory to transfer information from previous context windows into the current window, facilitating long-context understanding. Similarly, the in-context autoencoder (ICAE) [278] generates memory slots that accurately and comprehensively represent the original context, while LLMLingua [336, 337], Gist [279], and CompAct [280] further optimize long-prompt compression to reduce input context length.

Finally, *hallucination mitigation* strategies are essential for ensuring the reliability of generated outputs. These strategies aim to minimize the LLM’s tendency to produce factually incorrect or nonsensical content. One approach is implementing fact-checking mechanisms [338], verifying generated content against established knowledge or memory stores. Another involves uncertainty estimation [339, 340], where the model evaluates the confidence level of its generated content and flags or filters out low-confidence outputs. Additionally, knowledge-based decoding strategies can

be employed during the generation phase, introducing constraints that guide the model towards more factually accurate content. These techniques collectively contribute to generating more trustworthy outputs and aligned with the agent’s established knowledge base. Recent research has introduced expert memory subnetworks, such as PEER [283] and Lamini Memory Tuning [281], which specialize in memorizing specific types of information, including world knowledge and AI agents’ past experiences. These subnetworks offload memorization to dedicated parameters, reducing the main model’s propensity to hallucinate. By implementing these memory utilization strategies, agents can become more capable, accurate, and reliable. They can successfully leverage their memory stores to achieve superior performance across complex tasks.

3.5 Summary and Discussion

The development of truly intelligent agents depends not just on robust memory systems, but also on their seamless integration with other cognitive functions like perception, planning, reasoning, and action selection. Memory is not an isolated module; it is deeply intertwined with these other processes. For example, sensory input is encoded and filtered before storage (as discussed in the sections on memory representation and lifecycle), highlighting the interplay between perception and memory. Long-term memory, especially procedural memory, directly informs action selection through learned skills and routines. Retrieval mechanisms, like context-aware semantic similarity computation, are crucial for planning, allowing agents to access relevant past experiences. This interplay extends to the concept of a “world model.”

Central to intelligent agents is their ability to build and utilize internal world models. These models, representing an agent’s understanding of its environment, enable simulation, reasoning about consequences, and prediction. Robust world models are crucial for higher-level cognition, planning, and human-like intelligence. A world model is, in essence, a highly structured, often predictive, form of long-term memory. Memory provides the raw material—knowledge and experiences—for constructing the world model, while the world model, in turn, acts as an organizing framework, influencing how new memories are encoded, consolidated, and retrieved. For instance, a well-developed world model might prioritize storing surprising events, as these indicate gaps in the agent’s understanding.

However, developing effective world models and memory systems presents significant challenges. These include managing the complexity of real-world environments, determining the appropriate level of abstraction (balancing accuracy, complexity, and computational efficiency), and integrating multi-modal information. Learning and updating these models efficiently, avoiding bias, ensuring generalization, and enabling continuous adaptation are also critical. Furthermore, model-based planning requires efficient search algorithms to handle the inherent uncertainty in the model’s predictions.

Future research should focus on enhancing agent memory systems by drawing inspiration from the strengths of human memory, particularly its flexibility, adaptability, and efficiency. While agent memory has advanced considerably, it still lags behind human memory in these key areas. Human memory is remarkably associative, retrieving information from incomplete or noisy cues, and it exhibits a sophisticated form of “forgetting” that involves consolidation and abstraction, prioritizing relevant information and generalizing from experiences. Agent memory, conversely, often relies on precise matching and struggles with ambiguity.

Several promising research directions emerge. Exploring biologically-inspired mechanisms, such as neural memory networks (as discussed earlier), could lead to significant breakthroughs. Another crucial area is developing memory systems that actively “curate” their contents—reflecting on information, identifying inconsistencies, and synthesizing new knowledge. This requires integrating metacognitive capabilities (monitoring and controlling one’s own cognitive processes) into agent architectures. Furthermore, creating more robust and nuanced forms of episodic memory, capturing not just the “what” and “when” but also the “why” and the emotional context of events, is essential for agents that can truly learn from experience and interact with humans naturally.

Overcoming these challenges requires innovative solutions at the intersection of deep learning, reinforcement learning, and cognitive science. Developing more sophisticated and adaptable world models and memory systems—ones that mirror the strengths of human cognition—will pave the way for agents with a deeper understanding of their environment, leading to more intelligent and meaningful interactions.

Chapter 4

World Model

A world model enables an agent to predict and reason about future states without direct trial-and-error in reality. This section explores how human cognitive studies on “mental models” relate to AI world models in artificial intelligence, categorizing them under four paradigms: *implicit paradigm*, *explicit paradigm*, *simulator-based paradigm*, and a class of other emergent methods (e.g., *instruction-driven paradigm*). We then discuss how world models inherently intersect with other agentic components and conclude with open questions and future directions that unite these perspectives under a unified theoretical and practical framework.

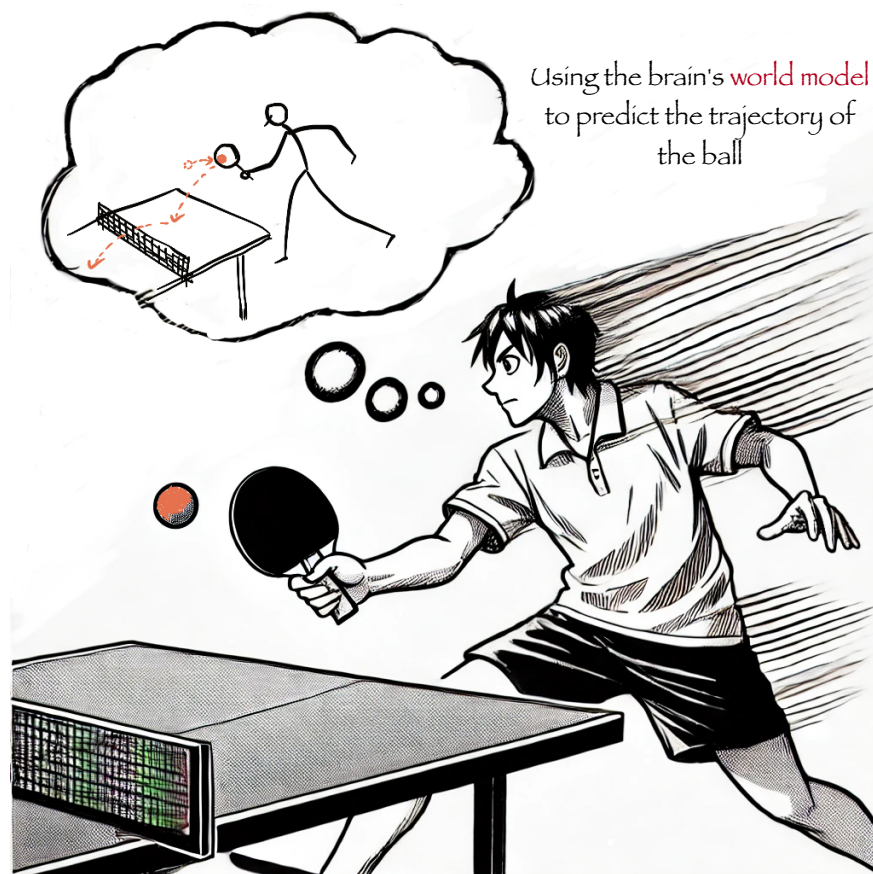


Figure 4.1: Humans can use their brain’s model of the world to predict the consequences of their actions. For example, when playing table tennis, a player can imagine or predict the trajectory of the ball after an action.

4.1 The Human World Model

Humans naturally construct internal representations of the world, often referred to as *mental models* in psychology [341, 342, 343]. These models serve as compact and manipulable depictions of external reality, enabling individuals to predict outcomes, plan actions, and interpret novel scenarios with minimal reliance on direct trial-and-error. Early work on spatial navigation, for instance, showed that humans and animals form “cognitive maps” of their surroundings [341], suggesting an underlying ability to imagine potential paths before actually traversing them.

Craik’s seminal argument was that the human mind runs internal “small-scale models of reality” [342] to simulate how events might unfold and evaluate possible courses of action. Later studies proposed that such simulations stretch across modalities—vision, language, and motor control—and are dynamically updated by comparing predictions to new observations. This process merges *memory recall* with *forward projection*, implying a close interplay between stored knowledge and the active generation of hypothetical future states [343]. More recent predictive processing theories such as “Surfing Uncertainty” [344] propose that the brain operates as a hierarchical prediction machine, continuously generating top-down predictions about sensory inputs and updating its models based on prediction errors.

Critically, these human mental models are:

- **Predictive:** They forecast changes in the environment, informing decisions about where to move or how to respond.
- **Integrative:** They combine sensory input, past experience, and abstract reasoning into a unified perspective on “what might happen next”.
- **Adaptive:** They are revised when reality diverges from expectation, reducing the gap between imagined and actual outcomes over time.
- **Multi-scale:** They operate seamlessly across different temporal and spatial scales, simultaneously processing immediate physical dynamics (milliseconds), medium-term action sequences (seconds to minutes), and long-term plans (hours to years). This flexibility allows humans to zoom in on fine-grained details or zoom out to consider broader contexts as needed.

Consider hunger and eating as an illustration of integrated world modeling. When hungry, a person’s internal model activates predictions about food—simulating not just visual appearance but tastes, smells, and anticipated satisfaction—triggering physiological responses like salivation before food is even present. This demonstrates seamless integration across perception, memory, and action planning.

The example also highlights adaptivity: once satiated, the same model dynamically updates, reducing predicted reward values for further eating. Despite recognizing the same food items, their anticipated utility changes based on internal state. Furthermore, humans maintain counterfactual simulations—declining dessert now while accurately predicting they would enjoy it later—enabling complex planning across hypothetical scenarios and time horizons, a capability comprehensive AI world models strive to replicate.

In sum, the *human world model* is not a static library of facts, but a flexible and ever-evolving mental construct, deeply rooted in perception and memory, that continuously shapes (and is shaped by) the individual’s interactions with the outside world.

4.2 Translating Human World Models to AI

Research in artificial intelligence has long sought to replicate the *predictive, integrative, and adaptive* qualities exhibited by human mental models [341, 342]. Early reinforcement learning frameworks, for instance, proposed learning an *environment model* for planning—exemplified by Dyna [345]—while contemporaneous work investigated using neural networks to anticipate future observations in streaming data [346, 347]. Both directions were motivated by the idea that an internal simulator of the world could enable more efficient decision-making than purely reactive, trial-and-error learning.

Subsequent advancements in deep learning brought the notion of “AI world models” into sharper focus. One influential approach introduced an end-to-end *latent generative model* of an environment (e.g., “World Models” [348]), whereby a recurrent neural network (RNN) and variational auto-encoder (VAE) together learn to “dream” future trajectories. These latent rollouts allow an agent to train or refine policies offline, effectively mirroring how humans mentally rehearse actions before executing them. Alongside such implicit designs, explicit forward-modeling methods emerged in model-based RL, letting agents predict $P(s' \mid s, a)$ and plan with approximate lookahead [349, 350].

Another branch of work leveraged large-scale simulators or real-world robotics to ground learning in richly diverse experiences [351, 352]. Such setups are reminiscent of how human children learn by actively exploring their environments, gradually honing their internal representations. Yet a key question lingers: can agentic systems unify these approaches (implicit generative modeling, explicit factorization, and simulator-driven exploration) into a cohesive “mental model” akin to that observed in humans? The recent proliferation of language-model-based reasoning [107, 74] hints at the potential to cross modalities and tasks, echoing how humans integrate linguistic, visual, and motor knowledge under one predictive framework.

Overall, as AI systems strive for flexible, sample-efficient learning, the *AI world model* stands as a conceptual bridge from cognitive theories of mental models to implementations that equip artificial agents with *imagination*, *predictive reasoning*, and *robust adaptation* in complex domains.

4.3 Paradigms of AI World Models

Designing an *AI world model* involves determining how an AI agent acquires, represents, and updates its understanding of the environment’s dynamics. While implementations vary, most approaches fall into four broad paradigms: *implicit*, *explicit*, *simulator-based*, and *hybrid or instruction-driven* models. These paradigms can be further analyzed along two key dimensions: reliance on *internal* (neural-based) vs. *external* (rule-based or structured) mechanisms, and overall *system complexity*. Figure 4.2 illustrates this two-dimensional space, showing how different approaches distribute themselves across these axes. Generally, implicit models tend to rely more on internal mechanisms, while explicit and simulator-based models incorporate more external structures. Simulator-based and explicit models also tend to be more complex than implicit and hybrid approaches, reflecting their structured reasoning and engineered constraints.

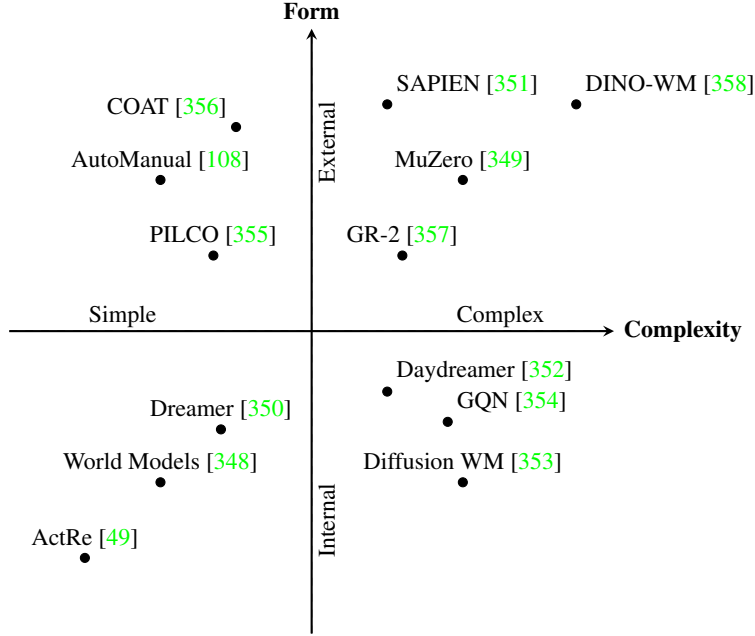


Figure 4.2: A two-dimensional layout of AI world-model methods. The horizontal axis indicates *Complexity* (left to right). The vertical axis spans *Internal* approaches (bottom) to *External* solutions (top). Approximate positions reflect each method’s reliance on large learned networks vs. explicit rules or code, and its overall system complexity.

4.3.1 Overview of World Model Paradigms

An *AI world model* is broadly any mechanism by which an agent captures or accesses approximate environment dynamics. Let \mathcal{S} denote the set of possible environment *states*, \mathcal{A} the set of *actions*, and \mathcal{O} the set of *observations*. In an idealized Markovian framework, the environment is characterized by transition and observation distributions:

$$T(s'|s, a) : \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S}), \quad (4.1)$$

$$O(o|s') : \mathcal{S} \rightarrow \Delta(\mathcal{O}), \quad (4.2)$$

where $T(\cdot)$ dictates how states evolve under actions, and $O(\cdot)$ defines how states produce observations. A **world model** typically *learns* or *utilizes* approximations of these functions (or a variant), allowing the agent to *predict* future states or observations without executing real actions in the environment.

Numerous approaches exist to implement these approximations, which we group into four main **paradigms**:

- **Implicit paradigm:** A single neural network or latent structure encodes both transition and observation mappings without explicit factorization. World Models [348] or large language models used for environment reasoning are typical examples. Agents generally unroll this black-box function to simulate hypothetical trajectories.
- **Explicit paradigm:** The agent directly models or has access to learnable transition model T_θ and observation model O_θ , often enabling interpretability or modular design. Model-based RL methods—like MuZero [349] or Dreamer [350]—learn or refine T_θ , planning in an approximated state space. Generative visual models such as [353, 358] fall under this category if they explicitly predict the next states or frames.
- **Simulator-Based paradigm:** Rather than approximating (4.1)–(4.2), the agent relies on an external simulator or even the physical world as the ground-truth. Systems like SAPIEN [351] or real-robot pipelines [352] can be seen as “native” environment models that the agent queries. Although no learned $T(\cdot)$ is required, the agent pays a cost in terms of runtime or real-world risks.
- **Other paradigms (Hybrid or Instruction-Driven):** Methods that defy simple classification. They may store emergent rules in textual form [108], refine implicit LLM knowledge into partial causal graphs [356], or combine external components with learned sub-modules. Such approaches highlight the evolving nature of world-model research, where instructions, symbolic rules, or on-the-fly structures can complement more traditional approximations.

Throughout the remainder of this subsection, we examine how each paradigm addresses (or circumvents) Equations (4.1) and (4.2), the trade-offs in interpretability and scalability, and their relative merits for different tasks ranging from text-based to high-dimensional embodied control.

4.3.2 Implicit Paradigm

In the **implicit** paradigm, an agent encodes all environment dynamics—including how states evolve and how observations are generated—within a single (or tightly coupled) neural model. Formally, one maintains a latent state h_t that is updated according to

$$h_{t+1} = f_\theta(h_t, a_t), \quad \hat{o}_{t+1} = g_\theta(h_{t+1}), \quad (4.3)$$

where f_θ subsumes the transition function $T(\cdot)$ (and part of $O(\cdot)$) from Eqs. (4.1)–(4.2), but without making these components explicit. A classic example is the *World Models* framework [348], in which a Variational Autoencoder (VAE) first compresses visual inputs into latent codes, and a recurrent network predicts the next latent code, effectively “dreaming” trajectories in latent space. Recent work also explores repurposing large language models (LLMs) for environment simulation in purely textual or symbolic domains [107, 74], although these models are not always grounded in strict time-series or physics-based data.

Because implicit models fuse the transition and observation mechanisms into one monolithic function, they can be elegantly trained end to end and unrolled internally for planning. However, they tend to be opaque: it is difficult to interpret how precisely the network captures domain constraints or to inject knowledge directly into any part of the transition. This can be advantageous for highly complex environments where a single large-capacity model can discover latent structure on its own, but it also risks brittleness under distribution shifts. Overall, the implicit paradigm is appealing for its simplicity and flexibility, but it can pose challenges when interpretability, explicit constraints, or fine-grained control of the dynamics are required.

4.3.3 Explicit Paradigm

The **explicit** paradigm instead factorizes the world model, often by learning or encoding a transition function $\hat{T}_\theta(s_{t+1} | s_t, a_t)$ and an observation function $\hat{O}_\theta(o_{t+1} | s_{t+1})$. This explicit separation makes it possible to query each function independently. For instance, one might draw samples from

$$\hat{s}_{t+1} \sim \hat{T}_\theta(s_t, a_t), \quad \hat{o}_{t+1} \sim \hat{O}_\theta(\hat{s}_{t+1}). \quad (4.4)$$

Model-based reinforcement-learning algorithms like MuZero [349] or Dreamer [350] exemplify this paradigm by refining a forward model for planning. Other explicit approaches prioritize fidelity in generating future frames, such as

Diffusion WM [353], which applies diffusion processes at the pixel level, or DINO-WM [358], which rolls out future states within a pretrained feature space.

By factorizing transitions and observations, explicit methods can be more interpretable and more amenable to debugging and domain-specific constraints. That said, they are still sensitive to model errors: if \hat{T}_θ deviates significantly from reality, the agent’s planning and decision-making can become ineffective. Many explicit systems still rely predominantly on internal (neural) representations, but they may integrate external planners (e.g., tree-search algorithms) to leverage the explicit transition structure. This blend of learned and symbolic components offers a natural way to incorporate human knowledge, while preserving the strengths of deep learning.

4.3.4 Simulator-Based Paradigm

In the **simulator-based** paradigm, the agent outsources environment updates to a simulator, effectively bypassing the need to learn \hat{T}_θ from data. Formally,

$$(s_{t+1}, o_{t+1}) \leftarrow \mathcal{SIM}(s_t, a_t), \quad (4.5)$$

where \mathcal{SIM} is often an external physics engine or the real world itself. Platforms like SAPIEN [351] and AI Habitat provide deterministic 3D physics simulations, allowing agents to practice or iterate strategies in a controlled environment. Alternatively, methods such as Daydreamer [352] treat real-world interaction loops like a “simulator,” continually updating on-policy data from physical robots.

This approach yields accurate transitions (assuming the simulator accurately reflects reality), which alleviates the risk of learned-model errors. However, it can be computationally or financially expensive, especially if the simulator is high fidelity or if real-world trials are time-consuming and risky. As a result, some agents combine partial learned dynamics with occasional simulator queries, aiming to balance accurate rollouts with efficient coverage of state-action space.

4.3.5 Hybrid and Instruction-Driven Paradigms

Beyond these three primary paradigms, there is a growing number of **hybrid** or **instruction-driven** approaches, which blend implicit and explicit modeling or incorporate external symbolic knowledge and large language models. Often, these systems dynamically extract rules from data, maintain evolving textual knowledge bases, or prompt LLMs to hypothesize causal relationships that can then be tested or refined.

AutoManual [108], for example, iteratively compiles interactive environment rules into human-readable manuals, informing future actions in a more transparent way. Meanwhile, COAT [356] prompts an LLM to propose possible causal factors behind observed events, then validates or refines those factors via direct interaction, bridging text-based reasoning with partial learned models. Although these solutions offer remarkable flexibility—particularly in adapting to unfamiliar domains or integrating real-time human insights—they can be inconsistent in how they structure or update internal representations. As language-model prompting and real-time rule discovery continue to evolve, these hybrid methods are poised to become increasingly common, reflecting the need to balance end-to-end learning with the transparency and adaptability offered by external instruction.

Until now, we have introduced the four typical paradigms of existing world model techniques, as illustrated in Figure 4.3.5. As we can see, each type of technique has trade-offs in different aspects.

4.3.6 Comparative Summary of Paradigms

The table summarizes the key methods in AI world modeling, categorizing them based on their reliance on *external* or *internal* mechanisms, their complexity, and their respective paradigms. The form column uses \circ for external approaches and \bullet for internal ones, with mixed methods having both symbols. This classification aligns with the previous subsections, including the detailed discussion of each paradigm, and complements the visual representation in Figure 4.2.

4.4 Relationships to Other Modules

A comprehensive AI world model does not exist in isolation but interacts with several key components of the agent’s architecture. These include (but not limited to) the memory, perception, and action modules. In this subsection, we explore how world models integrate with these critical components to enable coherent and adaptive behavior in dynamic environments.

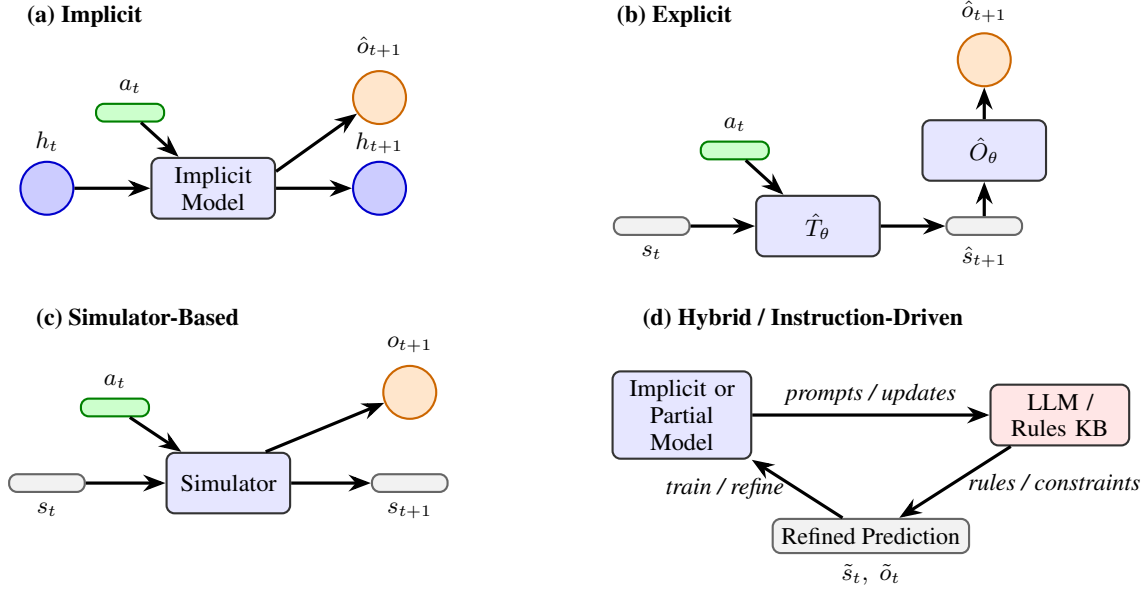


Figure 4.3: Four paradigms of world modeling: (a) implicit, (b) explicit, (c) simulator-based, and (d) hybrid/instruction-driven.

Table 4.1: Summary of AI world-model methods across paradigms, showing their form (External or Internal), complexity, and paradigm.

Method	Form	Complexity	Paradigm
ActRe [49]	•	Simple	Implicit
World Models [348]	•	Simple	Implicit
Dreamer [350]	•	Moderate	Implicit
Diffusion WM [353]	•	High	Explicit
GQN [354]	•	High	Explicit
Daydreamer [352]	○	High	Simulator-based
SAPIEN [351]	○	High	Simulator-based
PILCO [355]	○	Moderate	Explicit
AutoManual [108]	○	Simple	Other
MuZero [349]	○	High	Explicit
GR-2 [357]	•	High	Explicit
DINO-WM [358]	•	High	Explicit
COAT [356]	○	Moderate	Other

4.4.1 Memory and the World Model

Memory systems play a crucial role in the operation of world models. While a world model generates predictive representations of future states or actions, memory serves as the foundation upon which these representations are built and updated. The relationship between the world model and memory can be viewed as a loop where the world model predicts potential futures, while the memory stores past experiences, observations, and learned patterns, allowing for context-dependent reasoning and future predictions.

Memory mechanisms can be structured in various ways, including:

- **Short-term memory:** This enables the agent to hold and update its internal state temporarily, storing the most recent interactions or observations. This short-term context helps the agent make decisions in the immediate environment.
- **Long-term memory:** This serves as a more persistent repository of experiences and general knowledge about the environment. A world model can interact with long-term memory to refine its predictions, and it may use historical data to make more informed decisions or simulate more realistic futures.

For example, in model-based RL frameworks like Dreamer [350], recurrent neural networks act as both the world model and a form of memory, maintaining a latent state that is updated with each time step to predict future states. This form of integrated memory allows the agent to both recall past interactions and anticipate future ones.

4.4.2 Perception and the World Model

Perception refers to the agent’s ability to sense and interpret its environment through various modalities (e.g., vision, touch, sound, etc.). The world model relies heavily on accurate sensory input to form coherent predictions about the environment. In many AI systems, the perception module converts raw sensor data into a higher-level representation, such as an image, sound wave, or other structured data.

A key aspect of the interaction between the world model and perception is how the agent processes and integrates sensory input into the model. The world model often depends on processed data (such as features from convolutional neural networks or embeddings from transformers) to simulate potential futures. Additionally, the world model can guide perceptual processes by focusing attention on the most relevant sensory input needed to refine predictions.

For example, in autonomous robotics, perception systems typically detect objects or environmental features, which are then fed into a world model that predicts how the scene will evolve. RoboCraft [359] achieves this perception-to-modeling transformation by converting visual observations into particles and capturing the underlying system structure through graph neural networks. PointNet [360] further enriches perception systems’ understanding of physical space by encoding unstructured 3D point clouds to capture spatial characteristics of the environment. In navigation tasks, OVER-NAV [361] further combine large language models and open-vocabulary detection to construct the relationship between multi-modal signals and key information, proposing an omni-graph to capture the structure of local space as the world model for navigation tasks. This feedback loop between perception and the world model enables agents to update their perception dynamically based on ongoing predictions, allowing for real-time adaptation.

4.4.3 Action and the World Model

Action refers to the decision-making process through which an agent interacts with its environment. In agentic systems, actions are driven by the world model’s predictions of future states. The world model aids in planning by simulating the outcomes of different actions before they are executed, allowing the agent to choose the most optimal course of action based on the predicted consequences.

The integration between world models and action modules can take various forms:

- **Model-based planning:** World models explicitly model the environment’s transition dynamics [349, 362, 107], allowing the agent to simulate multiple action sequences (rollouts) before selecting the most optimal one.
- **Exploration:** World models also support exploration strategies by simulating unseen states or unexpected actions [363, 350, 364]. These simulations enable the agent to evaluate the potential benefits of exploring new parts of the state space.

In model-based planning, MuZero [349] performs implicit planning through self-play and Monte Carlo Tree Search (MCTS), transforming current state representations into future state and reward predictions to guide the decision-making process without prior knowledge of environment rules. In contrast, MPC [362] utilizes explicit dynamics models to predict multiple possible trajectories within a finite time horizon, determines the optimal control sequence by solving an optimization problem, and continuously updates planning using a receding horizon approach. Alpha-SQL [365], on the other hand, integrates an LLM-as-Action-Model within an MCTS framework to explore potential SQL queries within the database’s “world model”. This approach dynamically generates promising SQL construction actions based on partial query states, enabling zero-shot Text-to-SQL interactions without task-specific fine-tuning. Unlike MuZero, which focuses on planning for decision-making in uncertain environments, Alpha-SQL applies MCTS in a specific task—guiding SQL query construction through self-generated actions within a complex database context.

For exploration strategies, Nagabandi et al. [363] incentivizes agents to explore unknown regions by providing reward mechanisms (exploration bonuses) for discovering new states. Dreamer [350] propose that world models can generate imaginary action sequences (imaginary rollouts), allowing agents to safely evaluate the benefits of new actions in simulated environments without risking real-world experimentation. Similarly, in the discrete world model Hafner et al. [364], agents efficiently explore complex environments by simulating multiple possible future states, effectively balancing the trade-off between exploration and exploitation.

For example, in reinforcement learning, agents can employ a learned world model to simulate future trajectories in action-selection tasks. The world model evaluates the potential rewards of different actions, enabling the agent to plan effectively and take actions that maximize long-term goals.

4.4.4 Cross-Module Integration

While memory, perception, and action are discussed as separate modules, the true strength of world models lies in their ability to seamlessly integrate across these domains. A world model continuously receives sensory input, updates its internal memory, simulates future states, and uses this information to drive action selection. The iterative feedback loop between these modules allows agents to engage in intelligent, goal-directed behavior that is highly adaptive to changes in the environment.

This cross-module interaction is particularly relevant in complex, dynamic systems such as robotics, where an agent must continuously adapt its internal representation of the world, process sensory input, store relevant experiences, and take actions in real time. In the context of embodied agents, the integration of these modules ensures that predictions made by the world model are grounded in current observations and the agent’s ongoing experiences.

World models provide a fundamental unifying principle across modalities. Whether predicting physical outcomes in embodied robotics, anticipating visual changes on screens, or inferring semantic relationships in text, the core mechanism remains consistent: generating predictions about how states evolve under different actions. This cross-modal capacity explains why humans transition effortlessly between manipulating objects, navigating interfaces, and processing language—all activities driven by the same underlying predictive architecture. Future AI systems may achieve similar integration by developing world models that bridge these traditionally separate domains through a common predictive framework.

In summary, the relationship between the world model and the other modules—memory, perception, and action—forms the backbone of intelligent behavior in AI systems. Each module contributes to a cycle of prediction, update, and action, allowing agents to function effectively in dynamic and uncertain environments. These interactions highlight the need for a holistic approach when designing agent architectures, where world models are closely intertwined with sensory input, memory systems, and decision-making processes.

4.5 Summary and Discussion

The evolution of AI world models, from early cognitive insights to advanced AI architectures, underscores the growing realization that true intelligence relies on the ability to predict, simulate, and imagine. Unlike classical reinforcement learning, where agents operate solely through trial-and-error interactions, world models enable foresight—agents can plan, anticipate, and adapt to changes before they happen. This leap in cognitive modeling—whether implicit, explicit, or simulator-based—marks a significant shift in how machines can be endowed with flexibility, robustness, and generalization across tasks.

An essential yet often overlooked aspect of world models is their operation across multiple temporal and spatial scales. Human mental models seamlessly integrate predictions spanning milliseconds (reflexive responses), seconds (immediate action planning), minutes to hours (task completion), and even years (life planning) [366]. This multi-scale capability allows us to simultaneously predict immediate physical dynamics while maintaining coherent long-term narratives and goals. Similarly, humans process spatial information across scales—from fine-grained object manipulation to navigation across environments to abstract geographical reasoning. Current AI world models typically excel within narrow temporal and spatial bands, whereas human cognition demonstrates remarkable flexibility in scaling predictions up and down as context demands. This suggests that truly general-purpose AI world models may require explicit mechanisms for integrating predictions across multiple time horizons and spatial resolutions, dynamically adjusting the granularity of simulation based on task requirements.

One central challenge in designing world models is the interplay between **complexity** and **predictive accuracy**. As discussed, implicit models, such as those based on recurrent neural networks or transformers, offer simplicity and elegance, but they often come with the trade-off of limited interpretability. The model’s internal state is an opaque latent space, making it difficult to enforce domain constraints or provide guarantees about the accuracy of predictions. While such systems excel at capturing highly complex relationships and data-driven patterns, they also risk overfitting or failing to generalize to unseen scenarios.

Explicit models, by contrast, offer greater transparency and control. By factorizing state transitions and observations into separate functions, we gain a clearer understanding of how predictions are formed, and we can more easily integrate structured knowledge, such as physical laws or domain-specific rules. However, this approach comes with its own set of challenges. First, it often requires large amounts of labeled training data or simulated experiences to accurately capture environment dynamics. Second, even the most well-structured explicit models may struggle with complex environments that require fine-grained, high-dimensional state representations, such as in video prediction or robotics.

The **simulator-based** approach offers a promising alternative, wherein agents rely on external environments—either physically grounded or simulated—for dynamic updates. This method avoids many of the challenges inherent in learning accurate world models from scratch, as the simulator itself serves as the “oracle” of state transitions and observations. However, the reliance on simulators also introduces limitations: simulators often fail to capture the full richness of real-world dynamics and can be computationally expensive to maintain or scale. Furthermore, real-world environments introduce noise and variability that a purely learned or pre-configured model might miss. As AI agents strive to perform tasks in open-ended, unpredictable settings, the robustness of their world models will be tested by the gap between simulated and actual environments.

A key theme that emerges from this discussion is the **trade-off between generalization and specialization**. The more specific a world model is to a particular domain or task, the less likely it is to generalize across different contexts. Models like MuZero [349] and Dreamer [350] exemplify this: they excel at specific environments (e.g., Atari games or robotics) but require careful adaptation when transferred to new, uncharted domains. Conversely, implicit models—particularly those leveraging large-scale neural networks—have the potential to generalize across tasks but often do so at the cost of sacrificing domain-specific expertise.

Moreover, **integrating memory** with world models is crucial for agents that need to handle long-term dependencies and past experiences. While world models excel at predicting the next state based on immediate inputs, true intelligent behavior often requires reasoning about distant outcomes. Long-term memory allows agents to store critical environmental knowledge, ensuring that short-term predictions are grounded in a broader understanding of the world. This fusion of memory, perception, and action, mediated by the world model, creates a feedback loop where predictions shape actions, which in turn inform future predictions.

The **human analogy** remains compelling: just as humans integrate sensory inputs, memories, and internal models to navigate the world, so too must intelligent agents combine perception, memory, and action through their world models. As the field advances, it is clear that a holistic approach—one that unifies implicit, explicit, and simulator-based methods—may be the key to achieving more robust, generalizable, and adaptive agents. Hybrid methods, like those used in AutoManual [108] or discovery-based models [356], offer exciting possibilities for blending learned knowledge with structured rules and real-time interactions, potentially pushing the boundaries of what we consider a world model.

Looking forward, **open questions remain**. How can we ensure that world models exhibit **long-term stability** and **reliability** in real-world settings? How do we handle the inherent **uncertainty** in dynamic environments while maintaining the flexibility to adapt? Furthermore, as agents grow more sophisticated, how can we design systems that are both **efficient** and **scalable** across increasingly complex tasks without incurring massive computational costs?

In conclusion, the future of world models lies in their ability to balance the need for **generalization** with the requirement for **domain expertise**. By continuing to explore and refine the interplay between model simplicity and complexity, between external and internal approaches, we move closer to developing AI systems that not only understand the world but can actively shape their understanding to navigate and adapt in a rapidly changing reality.

Chapter 5

Reward

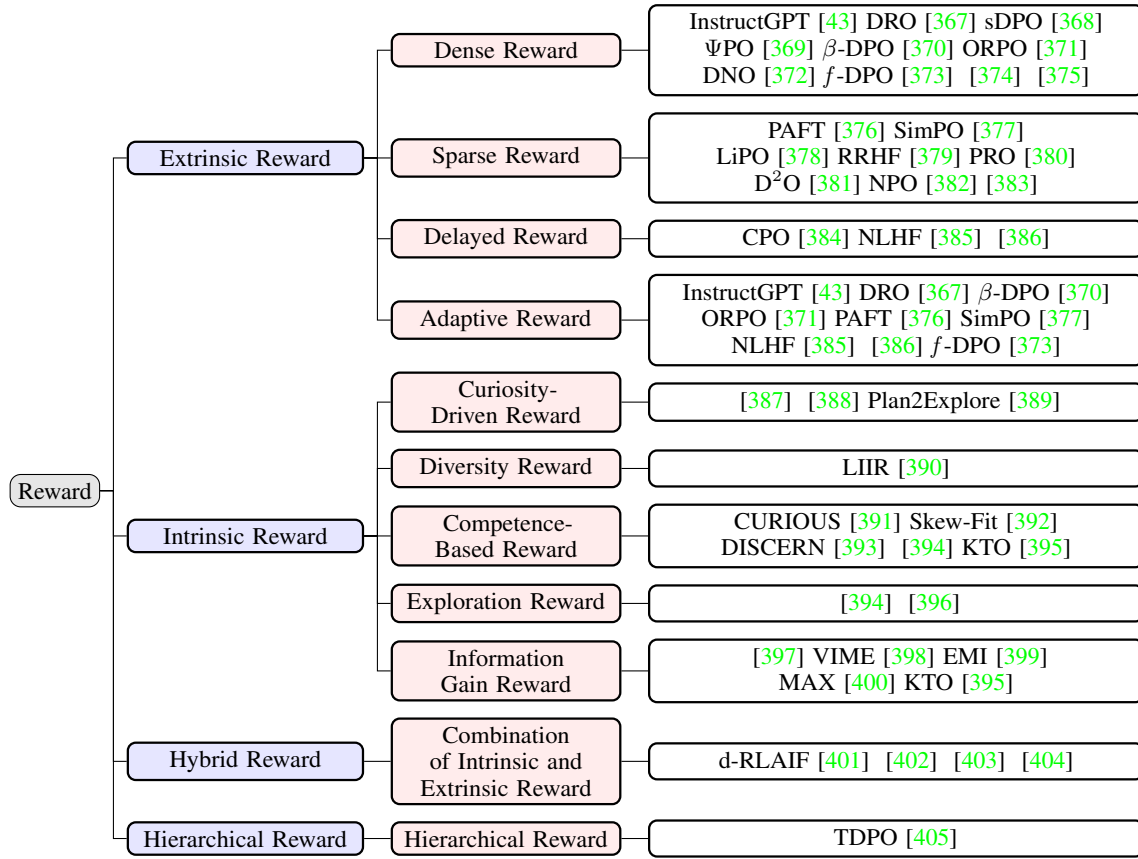


Figure 5.1: Illustrative Taxonomy of Reward system

Rewards help the agent distinguish between beneficial and detrimental actions, shaping its learning process and influencing its decision-making. This chapter first introduces common reward substances in the human body and the corresponding reward pathways. Then, the reward paradigm under the agent and the different methods involved are defined. In the discussion section, the influence relationship between other modules is described, and the existing methods are summarized, then the problems that need to be solved in the future and the optimization directions are discussed.

Table 5.1: The comparison of human common reward pathways.

Reward Pathway	Neurotransmitter	Mechanism
Mesolimbic pathway [406]	Dopamine	Dopaminergic neurons in the ventral tegmental area (VTA) extend projections to the nucleus accumbens, where they release dopamine to regulate reward-related signaling. Dopamine diffuses across the synaptic cleft and binds to dopamine receptors—primarily D1-like (excitatory via Gs proteins, increasing cAMP) and D2-like (inhibitory via Gi proteins, reducing cAMP)—thereby modulating reward, motivation, and reinforcement.
Mesocortical pathway [407]	Dopamine	Dopaminergic projections from the VTA reach the prefrontal cortex (PFC). Here, dopamine binds to its receptors to influence cognitive functions such as decision-making, working memory, and emotional regulation, all of which contribute to evaluating and anticipating rewards.
Nigrostriatal pathway [407]	Dopamine	Dopamine’s action on D1 and D2 receptors in the striatum helps shape both motor routines and reward-related behaviors.
Locus coeruleus [408]	Norepinephrine	Neurons in the locus coeruleus release norepinephrine to widely distributed targets across the brain. At synapses, norepinephrine binds to adrenergic receptors (α and β subtypes), modulating neuronal excitability, arousal, attention, and stress responses. These modulatory effects can indirectly influence reward processing and decision-making circuits.
Glutamatergic projection [409]	Glutamate	Upon releasing into the synaptic cleft, glutamate binds to both ionotropic receptors (such as AMPA and NMDA receptors) and metabotropic receptors located on the postsynaptic neuron, thereby initiating excitatory signaling. This binding produces excitatory postsynaptic potentials and is crucial for synaptic plasticity and learning within reward circuits.
GABAergic modulation [410]	Gamma-Aminobutyric Acid (GABA)	GABA serves as the principal inhibitory neurotransmitter. At the synapse, GABA binds to GABAA receptors and GABAB receptors. This binding results in hyperpolarization of the postsynaptic cell, thereby providing inhibitory regulation that balances excitatory signals in the reward network.

5.1 The Human Reward Pathway

The brain’s reward system is broadly organized into two major anatomical pathways. The first is the medial forebrain bundle, which originates in the basal forebrain and projects through the midbrain, ultimately terminating in brainstem regions. The second is the dorsal diencephalic conduction system, which arises from the rostral portion of the medial forebrain bundle, traverses the habenula, and projects toward midbrain structures [407]. The feedback mechanisms and substances in the human brain are complex, involving a variety of neurotransmitters, hormones, and other molecules, which regulate brain function, emotions, cognition, and behavior through feedback mechanisms such as neurotransmitter systems and reward circuits. Feedback mechanisms can be positive (such as feedback in the reward system) or negative (such as inhibiting excessive neural activity). Well-known feedback substances [411] include dopamine, neuropeptides, endorphins, glutamate, etc.

Dopamine is a signaling molecule that plays an important role in the brain, affecting our emotions, motivation, movement, and other aspects [412]. This neurotransmitter is critical for reward-based learning, but this function can be disrupted in many psychiatric conditions, such as mood disorders and addiction. The mesolimbic pathway [406], a key dopaminergic system, originates from dopamine-producing neurons in the ventral tegmental area (VTA) and projects to multiple limbic and cortical regions, including the striatum, prefrontal cortex, amygdala, and hippocampus. This pathway plays a central role in reward processing, motivation, and reinforcement learning, and is widely recognized as a core component of the brain’s reward system. Neuropeptides are another important class of signaling molecules in the nervous system, involved in a variety of functions from mood regulation to metabolic control, and are slow-acting signaling molecules. Unlike neurotransmitters, which are limited to synapses, neuropeptide signals can affect a wider range of neural networks and provide broader physiological regulation. There is a significant cortical-subcortical gradient in the distribution of different neuropeptide receptors in the brain. In addition, neuropeptide signaling has been shown to significantly enhance the structure-function coupling of brain regions and exhibit a specialized gradient from

sensory-cognitive to reward-physical function [413]. Table 5 lists the common reward pathways in the human brain, the neurotransmitters they transmit, and the corresponding mechanisms of action, describing the basic framework of the human brain reward system.

5.2 From Human Rewards to Agent Rewards

Having examined the foundations of human reward pathways, we now turn to how artificial agents learn and optimize behavior through reward signals. While biological systems rely on complex neurochemical and psychological feedback loops, artificial agents operate using formalized reward functions designed to guide learning and decision-making. Though inspired by human cognition, agent reward mechanisms are structurally and functionally distinct. Understanding the analogies and disanalogies between these systems is crucial for aligning artificial behavior with human preferences.

In humans, rewards are deeply embedded in a rich web of emotional, social, and physiological contexts. They emerge through evolutionarily tuned mechanisms involving neurotransmitters like dopamine and are shaped by experiences, culture, and individual psychology. In contrast, artificial agents rely on mathematically defined reward functions that are externally specified and precisely quantified. These functions assign scalar or probabilistic feedback to actions or states, providing a signal for optimization algorithms such as reinforcement learning [3, 414].

One key distinction lies in the programmability and plasticity of agent rewards. Unlike human reward systems, which are constrained by biological architecture and evolutionary inertia, agent reward functions are fully customizable and can be rapidly redefined or adjusted based on task requirements. This flexibility enables targeted learning but also introduces design challenges—specifying a reward function that accurately captures nuanced human values is notoriously difficult.

Another important disanalogy concerns interpretability and generalization. Human rewards are often implicit and context-dependent, whereas agent rewards tend to be explicit and task-specific. Agents lack emotional intuition and instinctual drives; their learning depends entirely on the form and fidelity of the reward signal. While frameworks like reinforcement learning from human feedback (RLHF) attempt to bridge this gap by using preference data to shape agent behavior [12], such methods still struggle with capturing the full complexity of human goals, especially when preferences are intransitive, cyclical, or context-sensitive [321].

Moreover, attempts to borrow from human reward mechanisms—such as modeling intrinsic motivation or social approval—face limitations due to the absence of consciousness, embodiment, and subjective experience in artificial agents. Consequently, while human reward systems offer valuable inspiration, the design of agent reward functions must address fundamentally different constraints, including robustness to misspecification, adversarial manipulation, and misalignment with long-term human interests.

The following section will delve deeper into agent reward models, focusing on their design principles, evolution, and how these models selectively incorporate human-inspired insights to optimize artificial behavior within formal systems.

5.3 AI Reward Paradigms

Rewards also exist in intelligent agents, especially in reinforcement learning scenarios. Rewards are the core signal used to guide how intelligent agents act in the environment. They express feedback on the behavior of intelligent agents and are used to evaluate an action’s quality in a certain state, thereby affecting the decision-making of subsequent actions. Through continuous trial and error and adjustment, intelligent agents learn to choose behavioral strategies that can obtain high rewards in different states.

5.3.1 Definitions and Overview

In reinforcement learning, the reward model dictates how an agent is provided with feedback according to the actions it performs within its environment. This model plays a crucial role in guiding the agent’s behavior by quantifying the desirability of actions in a given state, thus influencing its decision-making.

Formal Definition. The agent’s interaction with its environment can be framed within the formalism of a Markov Decision Process (MDP) [415], which is represented as:

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma), \quad (5.1)$$

where:

- \mathcal{S} denotes the state space, encompassing all possible states in the environment.
- \mathcal{A} denotes the action space, which encompasses all actions available to the agent at any given state.
- $P(s'|s, a)$ defines the state transition probability. It represents the likelihood of transitioning to state s' after the agent takes action a in state s .
- $r(s, a)$ specifies the reward function, which assigns an immediate scalar reward received by the agent for executing action a in state s .
- $\gamma \in [0, 1]$ is the discount factor, which controls the agent's preference for immediate versus future rewards by weighting the contribution of future rewards to the overall return.

The reward function $r(s, a)$ serves as a fundamental component in the formulation of the Agent Reward Model. It is mathematically represented as:

$$r(s, a) : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R} \quad (5.2)$$

This function returns a scalar reward based on the agent's current state s and the action a it selects. The scalar value $r(s, a)$ is a feedback signal that indicates the immediate benefit (or cost) of the chosen action in the given state. This reward signal guides the agent's learning process, as it helps evaluate the quality of actions taken within specific contexts.

Objective of the Agent Reward Model. The agent's primary objective is to maximize its overall cumulative reward over time. This is typically achieved by selecting actions that yield higher long-term rewards, which are captured in the form of the return G_t at time step t , defined as the sum of future discounted rewards:

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}, \quad (5.3)$$

where r_{t+k} denotes the reward received at time step $t + k$, and γ^k is the discount factor applied to rewards received at time step $t + k$. The agent aims to optimize its policy by maximizing the expected return over time.

At a higher level, the reward model can be classified into three categories based on the origin of the feedback signal: i) extrinsic reward, ii) intrinsic reward, iii) hybrid reward and iv) hierarchical model. Each of these categories can be further subdivided into smaller subclasses. Figure 5.2 illustrates different types of rewards. Next, we will explore these different types of reward in more detail, outlining the distinct features and applications of each type.

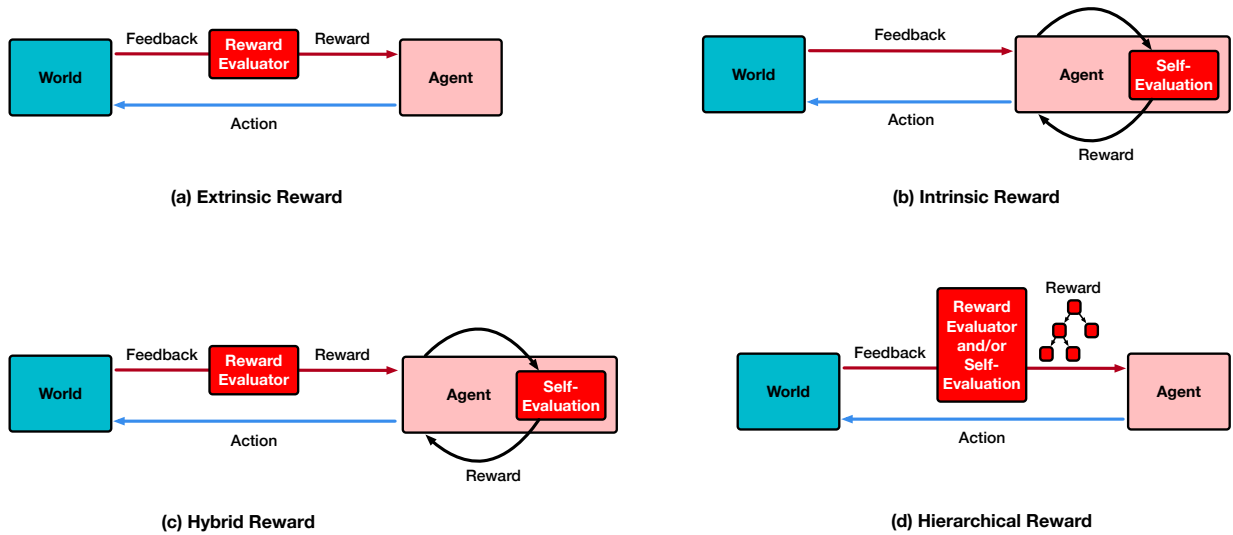


Figure 5.2: Illustration of different types of reward.

5.3.2 Extrinsic Rewards

Extrinsic rewards are externally defined signals that guide an agent’s behavior toward specific goals. In artificial learning systems, especially reinforcement learning, these signals serve as a proxy for success that shape the policy through measurable outcomes. However, the structure and delivery of these rewards significantly influence the learning dynamics, which present different trade-offs depending on how feedback is distributed.

Dense Reward. Dense reward signals provide high-frequency feedback, typically at every step or after each action. This frequent guidance accelerates learning by allowing agents to immediately associate actions with outcomes. However, dense feedback can sometimes incentivize short-sighted behavior or overfit to easily measurable proxies rather than deeper alignment.

For example, InstructGPT [43] uses human rankings of model outputs to provide continuous preference signals throughout fine-tuning, enabling efficient behavior shaping. Similarly, Cringe Loss [416] and its extensions [374] transform pairwise human preferences into dense training objectives, offering immediate signal at each comparison. Direct Reward Optimization (DRO) [367] further simplifies this paradigm by avoiding pairwise comparisons entirely, associating each response with a scalar score—making the reward signal more scalable and cost-effective. These methods exemplify how dense feedback facilitates fine-grained optimization but must be carefully designed to avoid superficial alignment.

Sparse Reward. Sparse rewards are infrequent and typically only triggered by major milestones or task completions. While they often reflect more meaningful or holistic success criteria, their delayed nature can make credit assignment more difficult, especially in complex environments.

PAFT [376] exemplifies this challenge by decoupling supervised learning and preference alignment, with feedback applied only at select decision points. This sparsity reflects a more global notion of success but increases the burden on optimization. Similarly, SimPO [377] uses log-probability-based implicit rewards without dense comparisons. The sparsity simplifies the training pipeline but can limit responsiveness to subtle preference shifts. Sparse reward systems thus tend to be more robust but demand stronger modeling assumptions or more strategic exploration.

Delayed Reward. Delayed rewards defer feedback until after a sequence of actions, requiring agents to reason about long-term consequences. This setup is essential for tasks where intermediate steps may be misleading or only make sense in retrospect. The challenge lies in attributing outcomes to earlier decisions, which complicates learning but encourages planning and abstraction.

Contrastive Preference Optimization (CPO) [384] trains models by comparing sets of translations rather than evaluating each one in isolation. The reward signal arises only after generating multiple candidates, reinforcing patterns across iterations. Nash Learning from Human Feedback [385] similarly delays feedback until the model identifies stable strategies through competitive comparisons. These methods leverage delayed rewards to push beyond surface-level optimization, aligning more with long-term goals at the cost of slower convergence and more complex training dynamics.

Adaptive Reward. Adaptive rewards evolve dynamically in response to the agent’s behavior or learning progress. By modulating the reward function such as increasing task difficulty or shifting reward targets, this approach supports continual improvement, especially in non-stationary or ambiguous environments. However, it introduces additional complexity in reward design and evaluation.

Self-Play Preference Optimization (SPO) [386] adapts rewards based on self-play outcomes, using social choice theory to aggregate preferences and guide learning. This approach allows the system to refine itself by evolving internal standards. f-DPO [373] builds on this idea by introducing divergence constraints that adapt the reward landscape during training. By tuning alignment-diversity trade-offs dynamically, these methods enable robust preference modeling under uncertainty, though they require careful calibration to avoid instability or unintended bias.

5.3.3 Intrinsic Rewards

Intrinsic rewards serve as internally generated signals that motivate agents to explore, learn, and improve, independent of external task-specific outcomes. These rewards are often structured to promote generalization, adaptability, and self-directed skill acquisition—qualities critical for long-term performance in complex or sparse-reward environments. Different intrinsic reward paradigms focus on fostering distinct behavioral tendencies within agents.

Curiosity-Driven Reward. This reward encourages agents to reduce uncertainty by seeking novel or surprising experiences. The key concept is to incentivize the agent to explore novel states where prediction errors are significant. This paradigm excels in sparse-reward settings by promoting information acquisition when external guidance is limited. For example, Pathak et al. [387] leverage an inverse dynamics model to predict the outcome of actions, creating a feedback loop that rewards novelty. Plan2Explore [389] extends this further by incorporating forward planning to

actively target areas of high epistemic uncertainty, thereby enabling faster adaptation to unseen environments. While effective at discovery, curiosity-driven methods can be sensitive to noise or deceptive novelty without safeguards.

Diversity Reward. Diversity reward shifts focus from novelty to behavioral heterogeneity, encouraging agents to explore a wide range of strategies rather than converging prematurely on suboptimal solutions. This approach is particularly useful in multi-agent or multimodal settings, where strategic variety enhances robustness and collective performance. LIIR [390] exemplifies this by assigning personalized intrinsic signals to different agents, driving them toward distinct roles while maintaining shared objectives. Diversity-driven exploration fosters broader policy coverage but may require careful balancing to avoid destabilizing coordination or goal pursuit.

Competence-Based Reward. Competence-based reward aims to foster learning progress by rewarding improvements in the agent’s task proficiency. This reward adapts dynamically as the agent grows more capable, which creates a self-curriculum that supports continual skill acquisition. Skew-Fit [392] facilitates this through entropy-based goal sampling, encouraging agents to reach diverse states while maintaining challenge. CURIOUS [391] further automates curriculum generation by selecting goals that maximize learning progress over time. Competence-based methods are well-suited for open-ended environments, though they often require sophisticated estimation of progress and goal difficulty.

Exploration Reward. Exploration reward directly incentivizes the agent to engage with under-explored states or actions, which emphasize breadth over depth in environment interaction. Unlike curiosity, which focuses on unpredictability, exploration reward often targets coverage or novelty relative to the agent’s visitation history. RND [394] exemplifies this by rewarding the prediction error of a randomly initialized network, pushing the agent toward unfamiliar states. This approach helps prevent premature convergence and encourages robustness, though it may lack focus if not paired with meaningful learning objectives.

Information Gain Reward. Information gain reward formalizes exploration as a process of uncertainty reduction, which guides agents to take actions that yield the highest expected learning. This reward is grounded in information theory and is especially powerful in model-based or reasoning-intensive tasks. CoT-Info [397] applies this to language models by quantifying knowledge gain at each reasoning step, optimizing sub-task decomposition. VIME [398] similarly employs Bayesian inference to reward belief updates about environmental dynamics. By explicitly targeting informational value, these methods offer principled exploration strategies, though they often incur high computational cost and require accurate uncertainty modeling.

5.3.4 Hybrid Rewards

Hybrid reward frameworks integrate multiple sources of feedback, most commonly intrinsic and extrinsic rewards, to enable more balanced and adaptive learning. By combining the exploratory drive of intrinsic rewards with the goal-directed structure of extrinsic rewards, these systems aim to improve both sample efficiency and generalization. This paradigm is especially beneficial in complex environments or open-ended tasks, where pure reliance on either feedback type may be insufficient.

A core advantage of hybrid rewards is their capacity to resolve the exploration-exploitation trade-off dynamically. For instance, Xiong et al. [403] combine intrinsic exploration with extrinsic human feedback within the context of RLHF. Using a reverse-KL regularized contextual bandit framework, they facilitate strategic exploration while aligning the agent’s actions with human preferences. The method integrates intrinsic and extrinsic rewards through an iterative DPO algorithm and multi-step rejection sampling, optimizing exploration and alignment without compromising efficiency.

5.3.5 Hierarchical Rewards

Hierarchical reward architectures decompose complex objectives into layered subgoals, each associated with distinct reward signals. This structure mirrors the hierarchical organization of many real-world tasks, allowing agents to coordinate short-term decisions with long-term planning. By assigning lower-level rewards to immediate actions and higher-level rewards to abstract goals, agents can learn compositional behaviors that scale more effectively to complex environments.

In language modeling, Token-level Direct Preference Optimization (TDPO) [405] illustrates this principle by aligning LLMs through fine-grained token-level rewards derived from preference modeling. Using forward KL divergence and the Bradley-Terry model, TDPO simultaneously refines local choices and global coherence, improving alignment with nuanced human preferences. The hierarchical reward process here is not merely a structural design but a functional one: reinforcing both micro-decisions and macro-outcomes in a coordinated fashion.

More generally, hierarchical rewards can serve as scaffolding for curriculum learning, where agents progressively learn from simpler subtasks before tackling the overarching objective. In LLM agents, this might mean structuring rewards for subcomponents like tool-use, reasoning chains, or interaction flows, each of which contributes to broader task success.

5.4 Summary and Discussion

5.4.1 Interaction with Other Modules

In intelligent systems, reward signals function not only as outcome-driven feedback but as central regulators that interface with core cognitive modules such as perception, emotion, and memory. In the context of LLM-based agents, these interactions become particularly salient, as modules like attention, generation style, and retrieval memory can be directly influenced through reward shaping, preference modeling, or fine-tuning objectives.

Perception. In LLM agents, perception is often realized through attention mechanisms that prioritize certain tokens, inputs, or modalities. Reward signals can modulate these attention weights implicitly during training, reinforcing patterns that correlate with positive outcomes. For example, during reinforcement fine-tuning, reward models may upweight specific linguistic features—such as informativeness, factuality, or politeness—causing the model to attend more to tokens that align with these traits. This parallels how biological perception prioritizes salient stimuli via reward-linked attentional modulation [417]. Over time, the agent internalizes a perception policy: not merely “what is said,” but “what is worth paying attention to” in task-specific contexts.

Emotion. Though LLMs do not possess emotions in the biological sense, reward signals can guide the emergence of emotion-like expressions and regulate dialogue style. In human alignment settings, models are often rewarded for generating responses that are empathetic, polite, or cooperative—leading to stylistic patterns that simulate emotional sensitivity. Positive feedback may reinforce a friendly or supportive tone, while negative feedback suppresses dismissive or incoherent behavior. This process mirrors affect-driven behavior regulation in humans [418], and allows agents to adapt their interaction style based on user expectations, affective context, or application domain. In multi-turn settings, reward-modulated style persistence can give rise to coherent personas or conversational moods.

Memory. Memory in LLM agents spans short-term context (e.g., chat history) and long-term memory modules such as retrieval-augmented generation (RAG) or episodic memory buffers. Reward signals shape how knowledge is encoded, reused, or discarded. For instance, fine-tuning on preference-labeled data can reinforce certain reasoning paths or factual patterns, effectively consolidating them into the model’s internal knowledge representation. Moreover, mechanisms like experience replay or self-reflection—where agents evaluate past outputs with learned reward estimators—enable selective memory reinforcement, akin to dopamine-driven memory consolidation in biological systems [419]. This allows LLM agents to generalize from prior successful strategies and avoid repeating costly errors.

In general, reward in LLM-based agents is not a passive scalar signal but an active agent of behavioral shaping. It modulates attention to promote salient features, guides stylistic and affective expression to align with human preferences, and structures memory to prioritize useful knowledge. As agents evolve toward greater autonomy and interactivity, understanding these cross-module reward interactions will be essential for building systems that are not only intelligent, but also interpretable, controllable, and aligned with human values.

5.4.2 Challenges and Directions

Although extensive research has been conducted on various reward mechanisms, several persistent challenges remain. One fundamental issue is reward sparsity and delay. In many real-world scenarios, reward signals are often infrequent and delayed, making it difficult for an agent to accurately attribute credit to specific actions. This, in turn, increases the complexity of exploration and slows down the learning process.

Another significant challenge is the potential for reward hacking. Agents, in their pursuit of maximizing rewards, sometimes exploit unintended loopholes in the reward function. This can lead to behaviors that diverge from the intended design goals, particularly in complex environments where optimization objectives may not always align with the true task requirements.

Moreover, the process of reward shaping presents a delicate balance. While shaping rewards can accelerate learning by guiding an agent toward desired behaviors, excessive or poorly designed shaping may lead to local optima, trapping the agent in suboptimal behaviors. In some cases, it may even alter the fundamental structure of the original task, making it difficult for the agent to generalize to other scenarios.

Many real-world problems are inherently multi-objective in nature, requiring agents to balance competing goals. Under a single reward function framework, finding the right trade-offs between these objectives remains an open problem. Ideally, a hierarchical reward mechanism could be designed to guide learning in a structured, step-by-step manner. However, constructing such mechanisms effectively is still a challenge.

Finally, reward misspecification introduces further uncertainty and limits generalization. Often, a reward function does not fully capture the true task goal, leading to misalignment between the agent's learning objective and real-world success. Additionally, many reward functions are tailored to specific environments and fail to generalize when conditions change or tasks shift, highlighting the need for more robust reward models.

Addressing these challenges requires novel approaches. One promising direction is to derive implicit rewards from standard examples or outcome-based evaluations, which can help mitigate reward sparsity issues. Additionally, decomposing complex tasks into hierarchical structures and designing rewards from the bottom up can offer a more systematic approach, even in multi-objective settings. Furthermore, leveraging techniques such as meta-learning and meta-reinforcement learning can enhance the adaptability of reward models, allowing agents to transfer knowledge across tasks and perform effectively in diverse environments. By exploring these avenues, we can move toward more reliable and scalable reward mechanisms that better align with real-world objectives.

Chapter 6

Emotion Modeling

Emotions are a key part of how humans think, make decisions, and interact with others. They guide us to understand situations, make choices, and build relationships. Antonio Damasio, in his book *Descartes' Error* [25], explained that emotions are not separate from logic. Instead, they are deeply connected to how we reason and act. When developing LLM agents, adding emotional capabilities can potentially make these systems smarter, more adaptable, and better understand the world around them.

For LLM agents, emotions can act as a decision-making tool, much like they do for humans. Emotions help us prioritize tasks, understand risks, and adapt to new challenges. Marvin Minsky, in *The Emotion Machine* [420], described emotions as a way to adjust our thinking processes, helping us solve problems in a more flexible and creative manner. Similarly, LLM agents with emotion-like features could improve their ability of solving complex problems and making decisions in a more human-style.

However, the integration of emotions into LLM agents is still in its early stages. Researchers are just starting to explore how emotional capabilities can improve these systems. Furthermore, there is great potential for LLM agents to support human emotional well-being, whether through empathetic conversations, mental health support, or simply building better connections with users. This promising but challenging area requires collaboration between fields such as psychology, cognitive science, and AI ethics. As research advances, emotion-understanding LLM agents could redefine how we interact with technology, creating deeper trust and more meaningful relationships between humans and machines.

In the following subsections, we will delve deeper into the role of emotions in shaping LLM agents. We will explore how emotions can be used to enhance learning and adaptability, how LLMs understand human emotions, and how these systems express and model their own emotional states. We will also examine how emotions can be manipulated to influence LLM agents' behavior and personalities, as well as the ethical and safety concerns that arise from these capabilities. Each of these discussions builds on the foundational importance of emotion to create LLM agents that are more intelligent, empathetic, and aligned with human values.

6.1 Psychological Foundations of Emotion

Psychological and neuroscientific theories of emotion provide essential frameworks for developing emotionally intelligent LLM agents. These theories can be categorized into several major approaches, each offering unique perspectives on how emotions function and how they might be implemented in AI systems.

Categorical Theories. These models posit that emotions exist as discrete, universal categories with distinct physiological and behavioral signatures. Ekman's theory of basic emotions [421] identifies six fundamental emotions (anger, disgust, fear, happiness, sadness, and surprise) that are recognized across cultures and expressed through specific facial configurations. This discrete approach has significantly influenced affective computing, with many emotion classification systems in AI adopting these labels for training [422, 423]. For LLM agents, categorical frameworks provide clear taxonomies for classifying user emotions and generating appropriate responses. However, they face criticism for oversimplifying the complex, blended nature of human emotional experience [424] and may not capture cultural variations in emotional expression [425].

Dimensional Models. Rather than discrete categories, dimensional approaches represent emotions as points in a continuous space defined by fundamental dimensions. Russell’s Circumplex Model [426] maps emotions onto two primary dimensions: valence (pleasure-displeasure) and arousal (activation-deactivation). This framework enables more nuanced tracking of emotional states. It distinguishes between high-arousal panic and low-arousal anxiety despite both having negative valence. The PAD (Pleasure-Arousal-Dominance) model [427] extends this by adding a dominance dimension, capturing the sense of control or power associated with emotional states. These continuous representations have proven valuable for LLM systems that need to generate emotionally graded responses or track subtle shifts in user affect over time [428, 429, 430]. Dimensional models allow for fine-grained control over generated content, enabling humans or agents to modulate tone along continuous scales rather than switching between discrete emotional states.

Hybrid and Componential Frameworks. Recognizing limitations in pure categorical or dimensional approaches, several theories integrate aspects of both. Plutchik’s Wheel of Emotions [431] arranges eight primary emotions in a wheel structure with intensity gradients and dimensional properties, allowing for the representation of complex emotional blends (e.g., love as a mixture of joy and trust). Meanwhile, componential models like Scherer’s Component Process Model (CPM) [432] conceptualize emotions as emerging from synchronized components including cognitive appraisal, physiological arousal, action tendencies, and subjective feelings. Particularly influential in AI research is the OCC (Ortony-Clore-Collins) model [433], which defines 22 emotion types based on how events, agents, or objects are evaluated relative to goals and standards. These appraisal-based frameworks have been implemented in dialogue systems that generate emotional responses through rule-based evaluation of situations [434, 435]. For LLM agents, such models provide computational structures for evaluating text input and selecting contextually appropriate emotional responses, improving both coherence and perceived empathy [436, 437].

Neurocognitive Perspectives. The neuroscience of emotion offers additional insights for LLM architectures. Damasio’s somatic marker hypothesis [25] emphasizes how emotions, implemented through body-brain interactions, guide decision-making by associating physiological states with anticipated outcomes. This interaction between the limbic system and the cortex shows a two-process architecture: fast “alarm” signals in the limbic system, like those processed by the amygdala, work alongside slower, more deliberate reasoning in the cortex. Contemporary LLM systems have begun implementing analogous architectures, where fast sentiment detection modules work in parallel with more thorough chain-of-thought reasoning [436, 437]. Recent evidence further suggests that opponent circuitry in the striatum enables distributional reinforcement learning by encoding not just mean rewards but entire probability distributions, offering a neural basis for emotion-influenced decision-making under uncertainty [438]. Similarly, LeDoux’s distinction between “low road” (quick, automatic) and “high road” (slower, cognitive) fear processing [24] suggests design patterns for systems that need both immediate safety responses and nuanced emotional understanding. Minsky’s framing of emotions as “ways to think” [420] that reorganize cognitive processes has influenced frameworks like EmotionPrompt [428] and Emotion-LLaMA [423], where emotional context dynamically reshapes LLM reasoning.

These theoretical frameworks increasingly inform the development of emotionally intelligent LLM agents. Categorical models provide clear labels for emotion classification tasks [423, 429], while dimensional embeddings enable continuous control over generated text [428]. Hybrid approaches help systems handle mixed emotions and emotional intensity. Appraisal-based methods, particularly those derived from the OCC model, allow LLMs to evaluate narrative events or user statements contextually, selecting appropriate emotional responses that foster rapport and trust [439]. Neuroscientifically-inspired dual-process architectures combine “fast” sentiment detection with “slow” deliberative reasoning, enabling both quick safety responses and deeper emotional understanding [436, 437]. While explicit neurocognitive mechanisms (like dedicated “amygdala-like” pathways) remain rare in current LLM pipelines, emerging research explores biologically-inspired modules to handle urgent emotional signals and maintain consistent emotional states across extended interactions [440, 441].

Emotion is a key part of human intelligence, and it will likely become one of the key components or design considerations of LLM agents. One key future direction is systematically translating these psychological and neuroscience theories into an LLM agent’s internal processes. Techniques for translating might include using dimensional models (e.g., valence/arousal/dominance) as latent states that influence generation or adopting explicit rule-based appraisals (OCC) to label user messages and shape the agent’s subsequent moves. Hybrid approaches offer a compelling balance: an LLM could first recognize a discrete category (e.g., “fear”) but also gauge its intensity and control dimension for finer-grained conversation. Such emotion-infused architectures might yield more coherent “moods” over time, analogous to how humans sustain affective states rather than resetting at every turn. Explicit alignment with psychological theories also enhances interpretability: designers can debug or refine the agent’s responses by comparing them to well-established emotion constructs, rather than dealing with opaque emergent behaviors.

A second direction is harnessing these theories to improve *affectionate or supportive interactions*, often referred to as emotional alignment. For example, circumplex or PAD-based tracking can help an LLM detect negative valence and high arousal in a user’s text and respond soothingly (e.g., lowering arousal, offering empathetic reappraisals). In

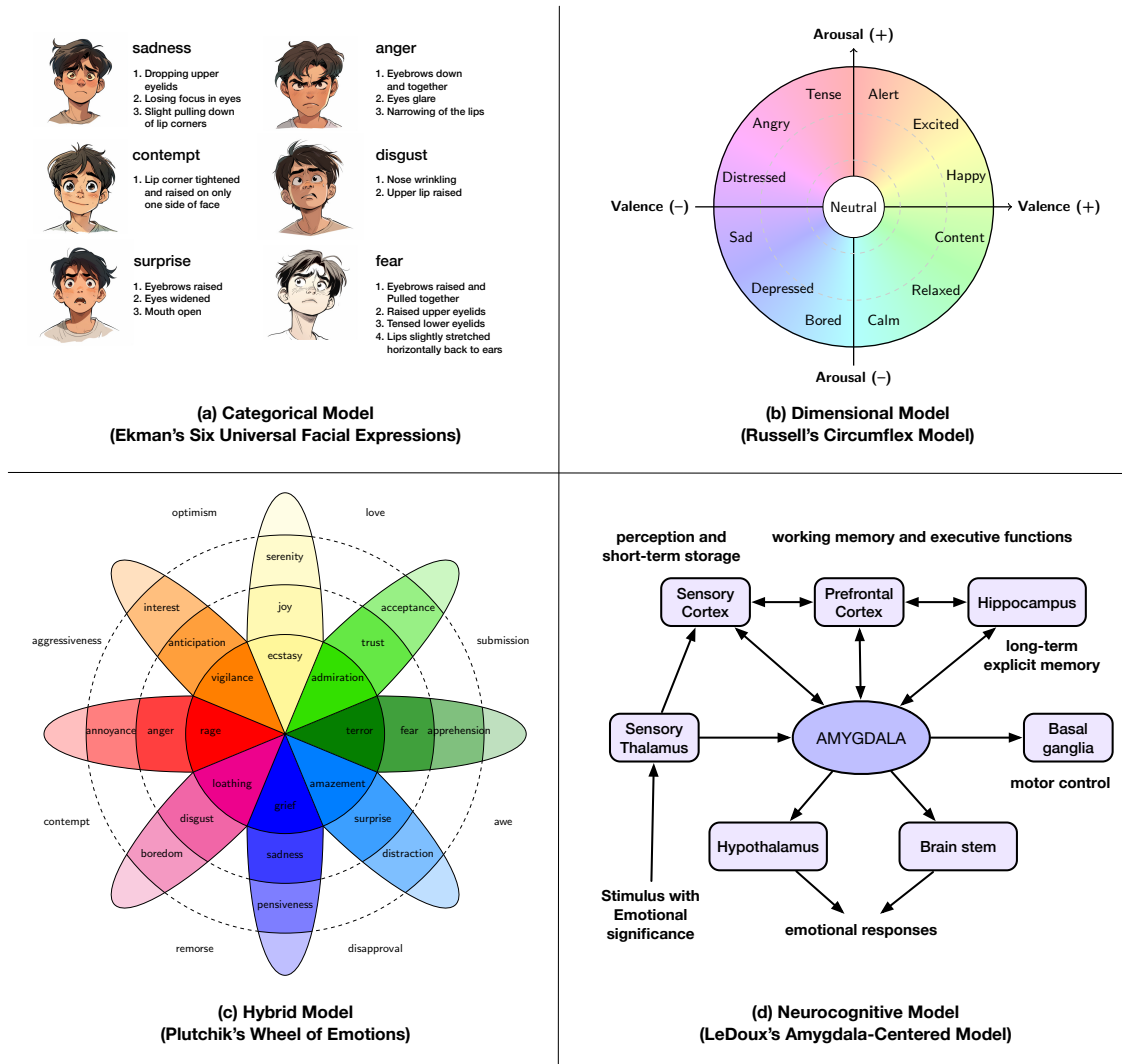


Figure 6.1: Visualization and examples of major emotion theory categories. (a) Categorical Theories: Ekman's six basic emotions [421] showing discrete emotional states. (b) Dimensional Models: Russell's Circumplex [426] representing emotions as coordinates in continuous space. (c) Hybrid/Componential Frameworks: Plutchik's Wheel [431] combining intensity gradients with categorical emotions. (d) Neurocognitive Perspectives: LeDoux's Amygdala-Centered Model [24] showing dual-pathway processing of emotional stimuli. These psychological foundations inform different approaches to emotion modeling in AI systems, from discrete classification to dimensional representations, appraisal-based reasoning, and multi-pathway information processing.

mental health or counseling scenarios, an appraisal-informed method could let the agent validate the user's feelings and understand their situation in terms of goal incongruence or perceived blame, which helps craft responses that convey genuine empathy. Grounding emotional outputs in cognitive theories (like "relief" if a negative outcome is avoided, or "gratitude" when a user helps the system) likewise makes interactions feel more natural and ethically aligned. These enhancements are particularly salient as LLMs migrate into real-world applications like customer service, elder care, and tutoring, where emotional sensitivity can improve outcomes and user well-being. By incorporating robust psychological and limbic-system insights, developers can design LLM agents that not only reason more effectively but also provide sincere emotional support, bridging the gap between computational precision and human-centric care.

6.2 Incorporating Emotions in AI Agents

The integration of emotional intelligence into large language models (LLMs) has emerged as a transformative approach to enhancing their performance and adaptability. Recent studies, such as those of EmotionPrompt [422], highlight how emotional stimuli embedded in prompts can significantly improve outcomes across various tasks, including a notable 10.9% improvement in generative task metrics such as truthfulness and responsibility. By influencing the attention mechanisms of LLMs, emotionally enriched prompts enrich representation layers and result in more nuanced outputs [422]. These advancements bridge AI with emotional intelligence, offering a foundation for training paradigms that better simulate human cognition and decision-making, particularly in contexts requiring social reasoning and empathy.

Multimodal approaches further elevate the impact of emotional integration. Models like Emotion-LLaMA [440] demonstrate how combining audio, visual, and textual data enables better recognition and reasoning of emotions. Using datasets such as MERR [440], these models align multimodal inputs into shared representations, facilitating improved emotional understanding and generation. This innovation extends beyond linguistic improvements, offering applications in human-computer interaction and adaptive learning. Together, these methods underscore the critical role of emotions in bridging technical robustness with human-centric AI development, paving the way for systems that are both intelligent and empathetic.

6.3 Understanding Human Emotions through AI

Textual Approaches. Recent work highlights the ability of LLMs to perform detailed reasoning about latent sentiment and emotion. Using step-by-step prompting strategies, such as chain of thought reasoning, researchers enable LLMs to infer sentiment even when explicit cues are absent [436]. Beyond single-turn inference, negotiation-based frameworks further refine emotional judgments by leveraging multiple LLMs that cross-evaluate each other's outputs, effectively mimicking a more deliberative human reasoning process [437]. These techniques underscore the importance of iterative, context-aware strategies to capture subtle emotional signals from purely textual input.

Multimodal Approaches. LLMs have also been extended to integrate signals from audio, video, and images. Recent efforts show how additional contextual or world knowledge can be fused with visual and textual information to capture deeper affective states [442]. Moreover, frameworks that convert speech signals into textual prompts demonstrate that vocal nuances can be embedded in LLM reasoning without changing the underlying model architecture [443]. This multimodal integration, combined with explainable approaches, allows for richer and more transparent representations of emotional content [444].

Specialized Frameworks. Beyond generic techniques, specialized systems address tasks in which emotion recognition requires higher levels of awareness of ambiguity [439], context sensitivity, and generative adaptability [445]. These approaches emphasize the inherent complexity of human emotion, treating it as dynamic and probabilistic rather than strictly categorical. Using flexible LLM instruction paradigms, they offer pathways to better interpret ambiguous emotional expressions and integrate contextual cues (e.g., dialogue history), moving LLM closer to human-like emotional comprehension.

Evaluation and Benchmarks. To holistically assess the emotional intelligence of LLM, researchers have proposed various benchmark suites. Some focus on generalized emotion recognition across different modalities and social contexts [446, 447], while others compare the performance and efficiency of models of varying sizes [448]. There are also specialized benchmarks that evaluate multilingual capabilities [449], annotation quality [450], or empathetic dialogue systems [451]. Furthermore, frameworks such as EMOBENCH [441] and MEMO-Bench [452] test nuanced emotional understanding and expression in both text and images, while MERBench [453] and wide-scale evaluations [454] address standardization concerns in multimodal emotion recognition. Together, these benchmarks reveal the growing, yet still imperfect grasp of human emotion by LLMs, highlighting ongoing challenges such as implicit sentiment detection, cultural adaptation, and context-dependent empathy [455].

6.4 Analyzing AI Emotions and Personality

Reliability of Personality Scales for LLMs. Large language models (LLMs) show conflicting evidence when evaluated through human-centered personality tests. On one hand, some studies challenge the validity of common metrics, reporting biases such as “agree bias” and inconsistent factor structures, raising doubts about whether these instruments capture genuine traits [456, 457]. On the other hand, systematic experiments reveal that LLMs can exhibit stable, human-like trait patterns and even adapt to different personas under specific prompts [458, 459]. Yet, concerns persist

about action consistency, alignment of self-knowledge, and whether role-playing agents truly maintain fidelity to their assigned characters [460, 461].

Psychometric Methods & Cognitive Modeling Approaches. Recent work applies rigorous psychometric testing, cognitive tasks, and population-based analyses to uncover how LLM processes and represents mental constructs [462, 463, 464]. Fine-tuning on human behavioral data can align models with decision patterns that mirror individual-level cognition, while population-based sampling techniques expose variability in neural responses [465, 466]. By merging psychological theories with advanced prompting and embedding methods, researchers illuminate latent representations of constructs like anxiety or risk-taking, showing how LLMs can approximate human reasoning across tasks.

Emotion Modeling. Studies on LLM-based emotional intelligence reveal notable abilities to interpret nuanced affect and predict emotion-laden outcomes, often surpassing average human baselines in standard tests [423, 429]. However, these models do not necessarily emulate human-like emotional processes; they rely on high-dimensional pattern matching that sometimes fails under changing contexts, negative input, or conflicting cues [467, 468]. However, hierarchical emotion structures, coping strategies, and empathy-like behaviors can emerge in larger-scale models, underscoring both the promise of emotional alignment and the ethical challenges in creating AI systems that appear and occasionally function as affective agents.

6.5 Manipulating AI Emotional Responses

Prompt-based Methods. Recent research shows that adopting specific personas or roles through well-engineered prompts can bias LLM cognition, allowing targeted emotional or personality outcomes [469, 470, 471, 472]. By inserting instructions such as “If you were a [persona]”, LLMs adapt not only their thematic style, but also their underlying emotional stance. This approach is powerful for real-time manipulation, though it can be inconsistent across tasks and model variants, highlighting the need for more systematic methods.

Training-based Methods. Fine-tuning and parameter-efficient strategies offer deeper, more stable ways to induce or alter LLM emotions [473, 428, 474]. Quantized Low-Rank Adaptation (QLoRA) and specialized datasets can embed nuanced traits such as the Big Five or MBTI profiles directly into the model’s learned weights. These methods enable LLMs to spontaneously exhibit trait-specific behaviors (including emoji use) and sustain their emotional states over longer dialogues, while also offering interpretability through neuron-level activation patterns.

Neuron-based Methods. A recent advance isolates personality-specific neurons and manipulates them directly to evoke or suppress emotional traits [475]. By toggling neuron activations pinpointed through psychologically grounded benchmarks (e.g., PersonalityBench), LLMs can embody targeted emotional dimensions without retraining the entire network. This neuron-centric approach provides fine-grained, dynamic control over model behaviors, representing a leap in precision and efficiency for emotional manipulation in LLMs.

6.6 Summary and Discussion

Manipulation and Privacy Concerns. The rapid adoption of Emotional AI in advertising and politics raises significant manipulation and privacy risks [476, 477]. Emotional AI often collects sensitive biometric data, such as facial expressions and voice tones, to infer emotional states, enabling targeted advertising or political influence. However, these systems can exploit human emotions for profit or political gain, infringing on fundamental rights and fostering over-surveillance in public spaces [478, 477]. Regulatory frameworks like GDPR and the EU AI Act are critical to mitigating these risks responsibly.

Alignment Issues. Emotional AI’s capacity to detect and interpret emotions is often misaligned with intended outcomes, leading to inaccuracies and biases. Anxiety-inducing prompts, for instance, have been shown to exacerbate biases in large language models (LLMs), affecting outputs in high-stakes domains such as healthcare and education [479, 480]. Misinterpretation of emotional cues by AI systems, as seen in workplace applications, can exacerbate discrimination and power imbalances [481]. Techniques like reinforcement learning from human feedback (RLHF) have proven effective in mitigating these issues but require further development to ensure robust alignment in diverse contexts [479, 423].

Ethical Implications. Trust and acceptance of AI systems are significantly influenced by their ability to exhibit empathy and maintain socially appropriate behavior [482, 483]. However, the commodification of emotions in workplace management and customer service has raised concerns about ethical labor practices and AI-human relationships [481]. Moreover, Emotional AI’s reliance on anthropomorphic characteristics without sufficient empathy can undermine user trust [482]. Frameworks like SafeguardGPT, which incorporate psychotherapy techniques, demonstrate promising approaches to fostering trust and aligning AI behavior with societal norms [484]. Nonetheless, challenges remain in ensuring privacy, fairness, and cultural sensitivity [484, 483].

Distinguishing AI Emotional Mimicry from Human Experience. Despite advances in emotion modeling for LLM agents, a fundamental distinction remains: these systems do not actually “feel” emotions as humans do but only show human-emotion-like patterns via probabilistic modeling. While LLMs can convincingly simulate emotional responses, recognize emotional patterns, and generate affectional outputs, they lack the embodied, phenomenological experience that defines human emotions. This simulation-reality gap creates both technical and ethical challenges. Users frequently anthropomorphize AI systems that display emotion-like behaviors [482], potentially leading to misplaced trust or expectations. This distinction needs to be carefully thought in both research and deployment contexts, as the perceived emotional capabilities of LLMs influence human-AI relationships, ethical frameworks, and regulatory approaches. Future work should balance enhancing LLMs’ emotional intelligence while maintaining transparency about their fundamental limitations as non-sentient systems.

Chapter 7

Perception

Perception is the foundational gateway through which both humans and intelligent agents acquire information, interpret their surroundings, and ultimately make informed decisions. For humans, perception is seamless and intuitive, effortlessly transforming sensory inputs into meaningful interpretations. In artificial intelligence, however, perception systems are meticulously engineered to emulate—and in some respects surpass—human sensory processing, profoundly influencing an agent’s capacity for interaction, learning, and adaptation in complex environments.

In this chapter, we begin by exploring key differences in the nature and efficiency of perception between humans and AI agents. Next, we categorize agent perception based on different forms and representations of perceptual input. We then discuss ongoing challenges in the agent perception system and highlight promising directions for improvement, both at the modeling and system architecture levels. Finally, we illustrate how perception modules can be effectively tailored to different intelligent agent scenarios, offering practical guidance for optimizing their use and suggesting pivotal areas for future research.

7.1 Human versus AI Perception

Perception is fundamental to intelligence, serving as the interface through which both humans and artificial agents interact with the world. Although humans commonly think of perception in terms of the five classical senses—vision, hearing, taste, smell, and touch—modern neuroscience identifies a richer sensory landscape. Conservatively, humans are described as having around 10 senses; more comprehensive views list approximately 21, while some researchers propose up to 33 distinct sensory modalities [546, 547]. Beyond the familiar senses, humans possess sophisticated internal perceptions, such as vestibular (balance), proprioception (awareness of body position), thermoception (temperature), and nociception (pain), enabling nuanced interaction with their environment.

Human senses are finely tuned to specific physical signals: for example, human vision detects electromagnetic waves with wavelengths between approximately 380–780 nm, whereas hearing perceives sound frequencies from about 20 Hz to 20 kHz [548]. These sensory modalities allow humans to effortlessly engage in complex tasks like language communication, object recognition, social interaction, and spatial navigation. Additionally, humans naturally perceive continuous changes over time, seamlessly integrating motion perception and temporal awareness, abilities essential for coordinated movement and decision-making [549]. Animals in the natural world exhibit even more diverse perceptual capabilities. Birds and certain marine organisms, for instance, utilize magnetoreception to navigate using Earth’s magnetic fields, while sharks and electric eels exploit electroreception to sense electrical signals emitted by other organisms—abilities humans do not possess [550].

In contrast to biological perception, artificial agents rely upon engineered sensors designed to transform environmental stimuli into digital signals that algorithms can interpret. Common sensor modalities for AI agents include visual sensors (cameras), auditory sensors (microphones), tactile sensors, and inertial measurement units. AI agents typically excel at processing visual, auditory, and textual data, leveraging advances in deep learning and signal processing. However, certain human sensory abilities—particularly taste and smell—remain challenging for machines to emulate accurately. For example, the advanced bio-inspired olfactory chip developed by researchers [551] currently distinguishes around 24 different odors, a capability significantly less sensitive than the human olfactory system, which discriminates among more than 4,000 distinct smells [552].

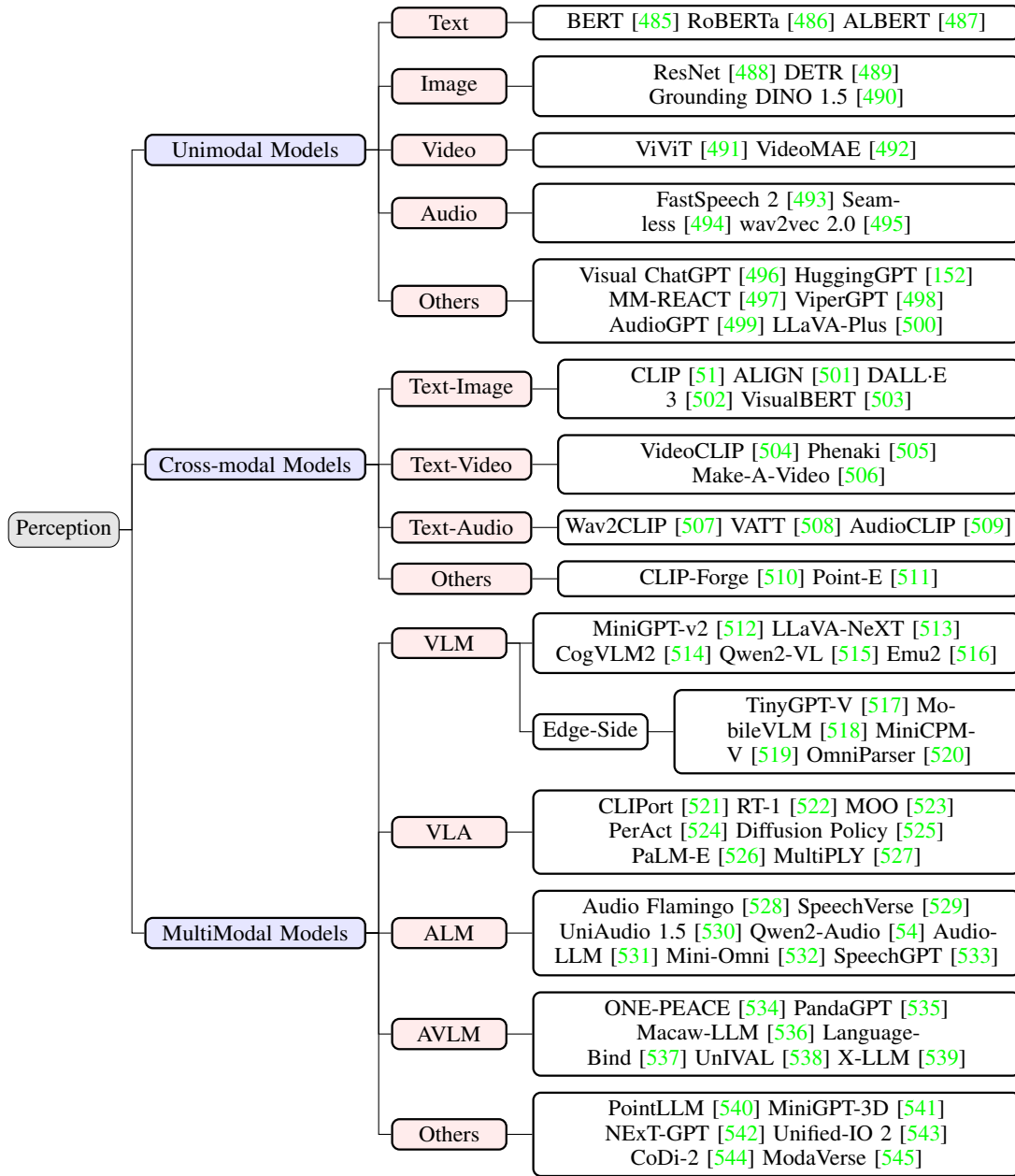


Figure 7.1: Illustrative Taxonomy of Perception System.

Another crucial distinction lies in perceptual processing efficiency. Human perception is limited by biological constraints such as nerve conduction speeds, typically in the range of milliseconds. Conversely, AI systems can process sensory inputs at speeds of microseconds or even nanoseconds, constrained primarily by computational hardware performance rather than biological limitations. Nevertheless, human perception naturally integrates information from multiple sensory modalities—known as multimodal perception—into coherent experiences effortlessly. For AI agents, achieving this multimodal integration requires carefully designed fusion algorithms that explicitly combine inputs from diverse sensors to build unified environmental representations [553].

Further differences arise in the way humans and artificial agents handle temporal and spatial information. Human perception is inherently continuous and fluid, smoothly experiencing the passage of time and spatial motion without explicit temporal discretization. In contrast, AI agents typically rely on discrete sampling of sensor data, using timestamps or sequential processing to simulate continuity. Spatial awareness in humans effortlessly merges visual, auditory, and vestibular information to achieve intuitive spatial positioning. For artificial agents, spatial perception usually involves

algorithmic processes such as simultaneous localization and mapping (SLAM) or 3D scene reconstruction from visual data sequences [554].

Physical or chemical stimuli transmitted from the external environment to human sensory organs will be received by the sensory system (such as eyes, ears, skin, etc.) and converted into neural signals, which are finally processed by the brain to produce perception of the environment. Similarly, to allow the intelligent agent to connect with the environment, it is also crucial to obtain these perception contents. Currently, various sensors are mainly used to convert electrical signals into processable digital signals. In this section, We distinguish between Unimodal models, Cross-modal models, and Multimodal models based on the number of modalities involved in the input and whether unified fusion modeling operations are performed. Unimodal Models specifically process and analyze data from a single modality or type of input (such as text, image, or audio), while Cross-modal Models establish relationships and enable translations between different modalities through dedicated mapping mechanisms, and Multimodal Models holistically integrate and process multiple modalities simultaneously to leverage complementary information for comprehensive understanding and decision-making.

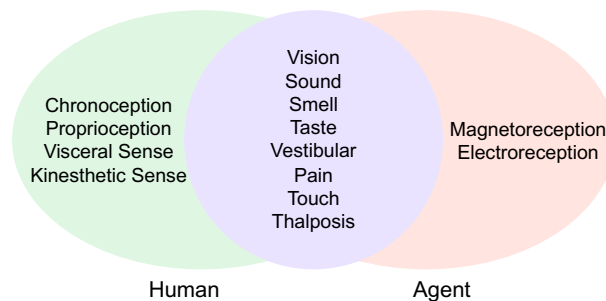


Figure 7.2: Comparison of common perceptual types between human and agent.

7.2 Types of Perception Representation

7.2.1 Unimodal Models

When humans are in an environment, they can listen to beautiful music, look at sunrise and sunset, or experience a wonderful audiovisual feast on stage. These perception contents can be either a single image or audio, or a fusion of multiple perception contents. Regarding the types of perception input of intelligent agents, we will start with single-modal and multimodal inputs, and introduce their implementation and differences.

Text As an important means of communication, text carries a wealth of information, thoughts, emotions and culture. Humans indirectly obtain the content of text through vision, hearing and touch, which is one of the most important ways for humans to interact with the environment. But for intelligent agents, text can directly serve as a bridge to connect with the environment, taking text as direct input and outputting response content. In addition to the literal meaning, text also contains rich semantic information and emotional color. In the early days, the bag-of-words model [555] was used to count text content and was widely used in text classification scenarios, but semantic expression could not be obtained. BERT [485] uses a bidirectional Transformer architecture for language modeling and captures the deep semantic information of text through large-scale unsupervised pre-training. [486, 487] further optimized the training efficiency of BERT. The autoregressive model represented by GPT3.5 [556] opened the prelude to LLM and further unified the tasks of text understanding and text generation, while technologies such as LoRA [109] greatly reduced the application cost of LLM and improved the agent’s perception ability of complex real-world scenario tasks.

Image Image is another important way for humans to interact with the environment which inherently encode spatial information, encompassing crucial attributes such as morphological characteristics, spatial positioning, dimensional relationships, and kinematic properties of objects. The evolution of computer vision architectures has demonstrated significant advancement in processing these spatial attributes. The seminal ResNet architecture [488] established foundational principles for deep visual feature extraction, while subsequent YOLO series [557, 558] demonstrated the capability to simultaneously determine object localization and classification with remarkable efficiency. A paradigm shift occurred with the introduction of DETR [489], which revolutionized object detection by implementing parallel prediction through global context reasoning, effectively eliminating traditional computational overhead associated with non-maximum suppression and anchor point generation. More recently, DINO 1.5 [490] has extended these capabilities to open-set scenarios through architectural innovations, enhanced backbone networks, and expanded training paradigms,

substantially improving open-set detection performance and advancing the perceptual generalization capabilities of artificial agents in unconstrained environments.

Video Video is an expression of continuous image frames, which includes the time dimension and displays dynamic information that changes over time through continuous image frames. The intelligent agent uses video as input and obtains richer perceptual content through continuous frames. ViViT [491] extracts spatiotemporal markers from videos, effectively decomposing the spatial and temporal dimensions of the input. VideoMAE [492] learns general video feature representations through self-supervised pre-training and has strong generalization capabilities on out-of-domain data. It lays a solid foundation for intelligent agents to acquire perceptual capabilities in new scenarios.

Audio In addition to text and vision, another important way for humans to interact with the environment is through audio. Audio not only contains direct text content, but also contains the speaker’s tone and emotion [559]. Wav2Vec2 [495] defines the contrast task by quantizing the potential representation of joint learning, achieving speech recognition effectiveness with 1/100 labeled data volume. FastSpeech 2 [493] directly introduces voice change information (pitch, energy, duration, etc.) and uses real targets to train the model to achieve more realistic text-to-speech conversion. Seamless [494] generates low-latency target translations through streaming and using an efficient monotonic multi-head attention mechanism, while maintaining the human voice style, to achieve synchronous speech-to-speech/text translation from multiple source languages to target languages. Based on these means, the intelligent agent can achieve the ability to listen and speak.

Others At present, most of the research on intelligent agents focuses on the above-mentioned common sensory input types. However, just as humans have more than 20 types of perception, intelligent agents have also made progress in achieving corresponding perception capabilities through other sensors. The bionic olfactory chip developed by Hong Kong University of Science and Technology [551] integrates a nanotube sensor array on a nanoporous substrate, with up to 10,000 independently addressable gas sensors on each chip, which is similar to the configuration of the olfactory system of humans and other animals, and can accurately distinguish between mixed gases and 24 different odors. In terms of taste, Tongji University [560] combines fluorescence and phosphorescence signals to develop an intelligent taste sensor with multi-mode light response, which can effectively identify umami, sourness and bitterness. In order to achieve human-like perception and grasping capabilities, New York University [561] launched a low-cost magnetic tactile sensor AnySkin, which can be quickly assembled and replaced. Even in the perception of pain, the Chinese Academy of Sciences uses the unique electrical properties of liquid metal particle films when they are “injured” (mechanically scratched) to imitate the perception and positioning of “wound.” Some other works, including HuggingGPT [152], LLaVA-Plus [500], and ViperGPT [498], integrate these single-modal perception capabilities within the framework, select and apply them according to task requirements, and achieve the goal of achieving more complex tasks.

7.2.2 Cross-modal Models

Text-Image Cross-modal models integrating text and images have witnessed significant advancements in recent years, leading to improved alignment, retrieval, and generation between the two modalities. These models can be categorized based on their primary objectives, including cross-modal alignment and retrieval, text-to-image generation, and image-to-text generation.

One of the primary focuses in cross-modal research is the alignment and retrieval of text and images. CLIP [51], introduced by OpenAI in 2021, employs contrastive learning to align textual and visual representations, enabling zero-shot cross-modal retrieval and classification. Similarly, ALIGN [501], developed by Google in the same year, leverages large-scale noisy web data to optimize text-image embedding alignment. In 2022, CyCLIP [562] introduced a cyclic consistency loss to further enhance the robustness of cross-modal alignment, improving the reliability of retrieval tasks.

Another major area of progress involves text-to-image generation, where models aim to synthesize high-quality images based on textual descriptions. OpenAI’s DALL-E series [563, 564, 502], spanning from 2021 to 2023, has made substantial contributions in this domain, with DALL-E 3 offering fine-grained semantic control over generated images. Stable Diffusion [565], introduced by Stability AI in 2022, employs a diffusion-based generative approach that supports open-domain text-to-image synthesis and cross-modal editing.

A third significant research direction is image-to-text generation, where models aim to generate high-quality textual descriptions based on image inputs. Typical representative work is the BLIP [566] and BLIP-2 [567] models, introduced by Salesforce between 2022 and 2023, which utilize lightweight bridging modules to enhance vision-language model integration, enabling tasks such as image captioning and question answering.

Text-Video The key research here involves video text alignment, generation and retrieval. VideoCLIP [504] employs a video encoder—typically based on temporal convolution or a transformer structure—to extract sequential features from video frames. These features are subsequently aligned with textual representations generated by a language encoder, facilitating robust video-text association. In the domain of text-to-video generation, Meta’s Make-A-Video model [506] extends spatial-temporal dimensions using diffusion-based techniques, allowing for high-quality video synthesis from textual descriptions. Additionally, Google’s Phenaki [505] addresses the challenge of generating long, temporally coherent video sequences, demonstrating significant advancements in video synthesis through cross-modal learning. DeepMind’s Frozen in Time [568] adopts contrastive learning for video-text matching, thereby enabling efficient cross-modal retrieval. This approach enhances the capacity to search and retrieve relevant video segments based on textual queries, further improving the integration of vision and language understanding.

Text-Audio Cross-modal models connecting text and audio have made significant improvements in related tasks such as modal representation, generation, and conversion, and enhanced the perception ability under a single modality.

AudioCLIP [509], introduced in 2021, extends the CLIP framework to the audio domain, enabling tri-modal retrieval across audio, text, and images. By incorporating audio as an additional modality, AudioCLIP utilizes multi-task learning to unify image, text, and audio representations into a shared embedding space. This advancement enhances the capability of cross-modal retrieval and interaction. In a similar vein, VATT [508] adopts a unified Transformer-based architecture to process video, audio, and text through independent encoding branches. These branches are subsequently fused into a shared multimodal space, facilitating tasks such as cross-modal retrieval and multi-task learning. This design allows for greater adaptability across diverse multimodal scenarios.

For text-to-audio generation, Meta introduced AudioGen [569] in 2023, which enables the synthesis of audio, such as environmental sounds and music fragments, directly from textual descriptions. This model exemplifies the growing capabilities of AI in generating high-fidelity audio based on linguistic input, expanding applications in media, entertainment, and accessibility.

Additionally, in the domain of speech-to-text and text-to-speech conversion, Microsoft developed SpeechT5 [570]. This model unifies speech and text generation, supporting both speech synthesis and recognition within a single framework. By leveraging a shared architecture for these dual functionalities, SpeechT5 contributes to the seamless integration of speech and text processing, thereby enhancing applications in automated transcription, voice assistants, and accessibility tools.

Others In some other scenarios and domains, cross-modal modeling also plays an important role.

CLIP-Forge [510] presents a novel method for generating 3D shapes from textual descriptions. By leveraging the capabilities of Contrastive Language-Image Pre-training (CLIP), this approach enables the synthesis of high-quality 3D objects conditioned on natural language inputs, bridging the gap between text and 3D geometry. Point-E [511] extends this concept by generating 3D point clouds from text descriptions. Unlike traditional 3D reconstruction techniques, Point-E focuses on point cloud representations, facilitating efficient and scalable 3D content creation while maintaining high fidelity to textual prompts.

In the field of medical imaging, MoCoCLIP [571] introduces an approach that enhances zero-shot learning capabilities. By integrating CLIP with Momentum Contrast (MoCo), this method improves the generalization of deep learning models in medical imaging applications, addressing the challenges associated with limited annotated data and domain adaptation.

7.2.3 Multimodal Models

The cross-modal model described above mainly aligns and maps between modalities through contrastive learning and other methods to achieve information complementarity and conversion between modalities. Furthermore, the work of multimodal models focuses on how to integrate the features of multiple data (such as vision, text, audio, etc.) to improve the performance of the overall model.

Vision Language Model Vision Language Model (VLM) is broadly defined as multimodal model that can learn from images(or videos) and text. Humans live in a world full of multimodal information. Visual information (such as images and videos) and language information (such as text) often need to be combined to fully express meaning. The same is true for intelligent agents. LLaVA [513] first tried to use gpt-4 to generate a multimodal language image instruction dataset. Through end-to-end training, a large multimodal model was obtained and excellent multimodal chat capabilities were demonstrated. LLaVA-NeXT [513] uses dynamic high-resolution and mixed data to show amazing zero-shot capabilities even in pure English modal data, and the computational/training data cost is 100-1000 times smaller than other methods. Emu2 [516] changes the traditional way of using image tokenizer to convert images into discrete tokens, and directly uses image encoders to convert images into continuous embeddings and provide them to Transformer,

enhancing multimodal context learning capabilities. MiniGPT-v2 [512] employs unique identifiers for various tasks during training. These identifiers help the model differentiate task instructions more effectively, enhancing its learning efficiency for each task. Qwen2-VL [515], DeepSeek-VL2 [572] use dynamic encoding strategies on visual components, aiming to process images with different resolutions and generate more efficient and accurate visual representations. At the same time, DeepSeek-VL2 [572] also uses the MoE model with a multi-head potential attention mechanism to compress the key-value cache into a latent vector to achieve efficient reasoning.

Previous work mainly uses image fusion text for training. Video-ChatGPT [573] extends the input to video and directly uses a video adaptive visual encoder combined with LLM for training to capture the temporal dynamics and inter-frame consistency relationships in video data, thereby enabling open conversations about video content in a coherent manner. To solve the lack of unified tokenization for images and videos, Video-LLaVA [574] unifies the visual representations of image and video encoding into the language feature space, making the two mutually reinforcing. Similarly, Chat-UniVi [575] employs a set of dynamic visual tokens to integrate images and videos, while utilizing multi-scale representations to allow the model to grasp both high-level semantic concepts and low-level visual details. Youku-mPLUG [576] has made in-depth research in specific scenarios. Based on the high-quality Chinese video-text pairs in the Youku video sharing platform, it enhances the ability to understand overall and detailed visual semantics and recognize scene text. Unlike the previous method that requires training, SlowFast-LLaVA [577] can effectively capture the detailed spatial semantics and long-term temporal context in the video through a two-stream SlowFast design without any additional fine-tuning of the video data, achieving the same or even better results than the fine-tuning method.

As the parameters of large models gradually decrease and the computing power of the end-side increases, high-performance end-side models are gaining momentum. Smart terminal devices such as mobile phones and PCs have strong demands for image visual processing, which puts forward higher multimodal recognition effects and reasoning performance requirements for the deployment of AI models on the end-side. TinyGPT-V [517] is built based on the Phi-2 [578] small backbone combined with BLIP-2 [567], only 8G video memory or CPU is needed for reasoning, and solving the computational efficiency problems of LLaVA [513] and MiniGPT-4 [579]. MiniCPM-V [519] mainly provides powerful OCR capabilities for long and difficult images, and has a low hallucination rate, providing reliable perception output. Megrez-3B-Omni [580] ensures that all structural parameters are highly compatible with mainstream hardware through coordinated optimization of software and hardware. Its inference speed is up to 300% faster than that of models with the same precision, improving its adaptability to different end-side hardware.

Similarly, there are more GUI-related works focusing on automatic task execution on mobile phones and PCs. Omni-Parser [520] uses popular web page and icon description datasets for fine-tuning, significantly enhancing the detection and functional semantic expression capabilities of icons in screenshots. GUICourse [581] and OS-ATLAS [582] also built a cross-platform GUI grounding corpus, which brought significant performance improvements in the understanding of GUI screenshots and enriching the interactive knowledge of GUI components.

Vision Language Action Model Vision-Language-Action (VLA) model, which takes vision and language as inputs and generates robotic actions as outputs, represents an important research direction in the field of embodied intelligence. The selection of vision and language encoders in VLA models has undergone diverse development, evolving from early CNNs to Transformer architectures, and further integrating 3D vision and large language models. Early models such as CLIPort [521] used ResNet [488] to process visual inputs and combined language embeddings to generate actions, laying the foundation for multimodal fusion. RT-1 [522] introduced the Transformer architecture, employing EfficientNet as the visual encoder and USE as the language encoder, and fused visual and language information via FiLM mechanisms, significantly enhancing the model’s generalization ability. VIMA [523] further adopted multimodal prompts, combining the ViT visual encoder and the T5 language model to support more complex tasks. PerAct [524] innovatively used 3D point clouds as visual inputs and processed multi-view information through Perceiver IO, providing richer spatial perception for robotic manipulation. Diffusion Policy [525] combined ResNet visual encoders and Transformer language models, generating actions through diffusion models to improve the diversity and accuracy of action generation. SayCan [583] integrated the PaLM language model with visual inputs, using the CLIP visual encoder for task decomposition. PaLM-E [526] combined the ViT visual encoder and the PaLM language model, guiding low-level action execution through text planning. MultiPLY [527] further integrated 3D information into LLMs, combining the EVA visual encoder and the LLaMA language model to provide more comprehensive planning capabilities for complex tasks.

Audio Language Model Audio Language Model(ALM) uses the audio and text to build multimodal model. Speechgpt [533] built a large-scale cross-modal speech instruction dataset SpeechInstruct and trained discrete speech representations, achieving cross-modal speech dialogue capabilities beyond expectations. LauraGPT [584], unlike the previous sampling of discrete audio tokens to represent input and output audio, proposed a novel data representation that combines the continuous and discrete features of audio, and demonstrated excellent performance on a wide range of

audio tasks through supervised multi-task learning. [529, 585, 531] converts audio data into embedded representations and then fine-tunes instructions, so that excellent performance can be achieved on various speech processing tasks through natural language instructions. In order to reduce the cost of fine-tuning training, Audio Flamingo [528] quickly enhances the ability to adapt to unseen tasks through contextual learning and retrieval based on the audio language model. UniAudio 1.5 [530] uses words or subwords in the text vocabulary as audio tokens, learns these audio representations through a small number of samples, and achieves cross-modal output without fine-tuning. In order to make the output more realistic and in line with human expectations, Qwen2-Audio [54] introduced the DPO training method to achieve human preference alignment.

Audio Vision Language Model Audio Vision Language Model (AVLM) utilizes audio, vision, and text to unify multimodal models. Previously, we introduced some work on building multimodal models using information from two modalities. In the pursuit of AGI, the obstacle to achieving this goal lies in the diversity and heterogeneity of tasks and modalities. A suitable approach is to allow more modal capabilities to be supported within a unified framework. Some closed-source work [586, 587] has achieved excellent capabilities across modalities such as text, vision, and audio. ImageBind [588] implements joint embedding across six different modes (image, text, audio, depth, thermal, and IMU data). Panda-GPT [535] combines ImageBind’s multi-modal encoder and Vicuna [589], showing zero-shot cross-modal performance in addition to images and text. Similar work includes [539, 539, 536], which achieves alignment and training through the encoding information of vision, audio and text. Multimodal models often require more resources to train, and UniVAL [538] trained a model with only $\sim 0.25B$ parameters based on task balance and multimodal curriculum learning, and used weight interpolation to merge multimodal models, maintaining generalization under out-of-distribution. NExT-GPT [542] connects LLM with multimodal adapters and different diffusion decoders, and only trains a small number of parameters (1%) of certain projection layers.

Other works [543, 590, 544, 545] have achieved input-output conversion between arbitrary modalities. Unified-IO 2 [543] is the first autoregressive multimodal model that can understand and generate images, text, audio, and actions. It tokenizes different modal inputs into a shared semantic space and processes them using an encoder-decoder model. AnyGPT [590] builds the first large-scale any-to-any multimodal instruction dataset, using discrete representations to uniformly process various modal inputs. Modaverse [545] directly aligns the output of the LLM with the input of the generative model to solve the problem that previous work relies heavily on the alignment of the latent space of text and non-text features, avoiding the complexity associated with the alignment of latent features. CoDi-2 [544] outperforms earlier domain-specific models in tasks like topic-based image generation, visual transformation, and audio editing.

Others Humans have explored the 2D world more than the 3D world, but 3D can more accurately describe the shape and texture information of objects and provide richer perceptual information. PointLLM [540] uses a point cloud encoder to express geometric and appearance features, and integrates language features for two-stage training of complex point-text instructions, achieving excellent 3D object description and classification capabilities. Since 3D contains richer information than 2D, it also brings greater training costs. [541, 591] reduces the training cost here, and MiniGPT-3D [541] uses 2D priors from 2D-LLM to align 3D point clouds with LLMs. Modal alignment is performed in a cascade manner, and query expert modules are mixed to efficiently and adaptively aggregate features, achieving efficient training with small parameter updates. LLaVA-3D [591] connects 2D CLIP patch features with their corresponding positions in 3D space, integrates 3D Patches into 2D LMM and uses joint 2D and 3D visual language command adjustment to achieve a 3.5-fold acceleration in convergence speed.

In order to enable intelligent agents to accurately perceive and manipulate unknown objects, Meta [592] developed NeuralFeels technology, which combines vision and touch to continuously model unknown objects in 3D, more accurately estimate the posture and shape of objects in handheld operations, and improve the accuracy of ignorant object operations by 94%.

7.3 Optimizing Perception Systems

Perception errors, including inaccuracies, misinterpretations, and “hallucinations” (generation of false information), pose substantial challenges to the reliability and effectiveness of LLM-based agents. Optimizing perception thus requires minimizing these errors using various strategies across model, system, and external levels.

7.3.1 Model-Level Enhancements

Fine-tuning. Fine-tuning pre-trained LLMs on domain-specific data significantly improves their ability to accurately perceive and interpret relevant information. For example, fine-tuning models such as LLaVA on specific landmarks has been shown to enhance their recognition accuracy, particularly in urban navigation tasks [513, 593]. Moreover, techniques such as Low-Rank Adaptation (LoRA) enable more efficient fine-tuning, avoiding a substantial increase in

model complexity while still improving performance [109, 594]. Some LLM work combined with traditional vision is also widely used. Integrating with YOLOs [595] on the basis of the the Llama-Adapter [596] architecture significantly improves the detection and positioning capability.

Prompt Engineering. The design of effective prompts is crucial to ensure LLMs generate outputs that are both accurate and aligned with the desired goals. By providing clear instructions, contextual information, and specific formatting requirements, prompt engineering minimizes misinterpretation and hallucination [597]. System prompts define the agent’s role, historical prompts to provide context from past interactions, and customized prompts to ensure output consistency has been shown to reduce errors significantly [597].

Retrieval-Augmented Generation. Supplementing LLMs with external knowledge sources through retrieval mechanisms helps ground their responses in factual information, reducing the likelihood of hallucinations and improving the accuracy of perceived information [334].

7.3.2 System-Level Optimizations

Anticipation-Reevaluation Mechanism. In scenarios where agents face incomplete or ambiguous information, an anticipation-reevaluation mechanism can enhance robustness. For instance, in navigation tasks, agents can anticipate goal directions based on historical data and reevaluate their inferences when new information becomes available [598].

Multi-Agent Collaboration. In multi-agent systems, structured communication and collaboration among agents can facilitate information sharing, error correction, and consensus-building, leading to a more accurate collective perception of the environment [599]. Different communication topologies, such as fully connected, centralized, and hierarchical structures, offer varying trade-offs in terms of efficiency and robustness [600]. InsightSee [601] refines visual information through a multi-agent framework with description, reasoning, and decision-making, effectively enhancing visual information processing capabilities. Similarly, HEV [602] integrates the global perspective information of multiple agents and endows RL agents with global reasoning capabilities through cooperative perception, thereby enhancing their decision-making capabilities.

Agent Specialization. Assigning distinct roles and capabilities to individual agents within a multi-agent system allows for a division of labor in perception, with each agent focusing on specific aspects of the environment or task. This can enhance the overall accuracy and efficiency of perception [603].

7.3.3 External Feedback and Control

Loss Agents for Optimization. Utilizing LLMs as loss agents, allows for the dynamic adjustment of loss function weights during training [604]. This enables the optimization of image processing models based on complex, potentially non-differentiable objectives, including human feedback and evaluations from specialized models. This approach essentially externalizes the optimization objective, allowing the LLM to “perceive” and adapt to complex criteria [605].

Human-in-the-Loop Systems. Incorporating human feedback and oversight can help correct errors, guide the agent’s learning process, and ensure alignment with human values and expectations [43].

Content and Output Mediation. Before presenting LLM outputs to users, content mediation filters and refines these outputs. This helps prevent unexpected or harmful behaviors, ensuring alignment with user expectations and safety guidelines [606].

7.4 Perception Applications

The operational efficacy of intelligent agents is predominantly influenced by three critical factors: model architecture dimensionality, hardware infrastructure specifications, and quantization optimization methodologies. The exponential progression in model parameters—from Bert-Base’s modest 110M to GPT-3’s substantial 175 billion, culminating in Llama 3’s unprecedented 405 billion—has correspondingly escalated processing latency from milliseconds to hundreds of milliseconds. Hardware performance variations are particularly noteworthy; empirical evidence with GPT-3 demonstrates that NVIDIA H100 exhibits a 50% improvement in token processing throughput compared to A100, while RTX 4090 achieves approximately double the processing capability.

Contemporary intelligent agents have penetrated diverse domains, encompassing personal assistance systems, gaming environments, Robotic Process Automation (RPA), and multimedia content generation, predominantly leveraging visual perception as their primary input modality. In the context of procedurally generated environments like Minecraft, STEVE [607] demonstrates remarkable performance improvements, achieving a 1.5x acceleration in technology tree progression and a 2.5x enhancement in block search efficiency through visual information processing. Steve-Eye [608]

advances this paradigm through end-to-end multimodal training, addressing environmental comprehension latency through integrated visual-textual input processing.

In creative content generation, AssistEditor [609] exemplifies sophisticated multi-agent collaboration, facilitating professional video editing through style-driven content understanding. Similarly, Audio-Agent [610] implements cross-modal integration between textual/visual inputs and audio outputs, enabling comprehensive audio manipulation capabilities [611, 612, 613].

Mobile and desktop platforms have witnessed significant advancements in agent applications. ExACT [614] has established new state-of-the-art benchmarks in VisualWebArena [615], achieving a 33.7% Success Rate through screenshot-based exploratory learning with caption and Set of Mask integration. SPA-Bench [616] introduces a comprehensive mobile evaluation framework that authentically replicates real-world complexity. M3A [617] demonstrates superior performance with a 64.0% success rate in SPA-Bench through multimodal input processing. AgentStore [618] has markedly improved OSWorld PC benchmark performance to 23.85% through enhanced visual and accessibility tree processing.

Voice interaction capabilities [619, 586] in personal AI assistants have significantly reduced interaction friction while enhancing operational efficiency. The integration of emotional prosody in voice interactions has demonstrated increased user engagement and retention.

In embodied intelligence applications, haptic and force feedback mechanisms have emerged as crucial modalities for environmental interaction, with enhanced sensory fidelity enabling increasingly precise operational capabilities [620].

7.5 Summary and Discussion

Although more and more research works [543, 590] focus on building unified multimodal models to support the input and output of multiple perception capabilities. Agent perception, a cornerstone of autonomous systems, faces significant challenges in effectively interpreting and integrating multi-modal data. Current methodologies encounter persistent issues in representation learning, alignment, and fusion, which hinder the development of robust and generalizable perception systems.

One of the primary issues lies in the representation methods employed, which often fail to capture the intricate nuances of multi-modal data. This shortfall is particularly evident in scenarios where high-dimensional sensory inputs require a sophisticated abstraction that preserves critical semantic information. Furthermore, the alignment of representations presents additional difficulties. Integrating heterogeneous data types into a cohesive feature space is not only computationally intensive but also prone to inconsistencies, which can lead to misinterpretation of ambiguous signals. The challenge is compounded when attempting to fuse these diverse representations, as the process of merging features from various sources frequently results in suboptimal integration and potential loss of vital information.

Future research directions should prioritize adaptive representation learning through dynamic neural architectures capable of automatically adjusting their structure based on environmental context and task demands. This could involve meta-learned parameterization or graph-based representations that explicitly model relationships between perceptual entities. For cross-modal alignment, self-supervised spacetime synchronization mechanisms leveraging contrastive learning principles show promise in establishing dense correspondence without requiring exhaustive labeled data. The integration of causal inference frameworks into alignment processes [621] could further enhance robustness against spurious correlations. In representation fusion, hierarchical attention mechanisms with learnable gating functions merit deeper exploration to enable context-aware integration of complementary modality features. Emerging techniques in differentiable memory networks may provide new pathways for maintaining and updating fused representations over extended temporal horizons.

Chapter 8

Action Systems

In the realm of philosophy, action is defined as the behaviors that agents can perform for a potential or specific purpose in the environment. For example, manipulation, moving, reasoning, and tool utilization can all be considered as fundamental actions that an intelligent agent can execute to fulfill a goal in real-world scenarios. In other words, actions emerge from the goal-oriented engagement of an agent in its environment, reflecting its intent to transform the external world in pursuit of its goals. Therefore, the action system also plays a vital role in differentiating AI agents and foundation models (e.g., LLMs). Generally, existing foundation models have demonstrated impressive performance across various tasks, but their task scope is still limited as they predominantly relies on the original pre-training objective (e.g., next-token prediction). By serving foundation models as brain intelligence, AI agents equipped with action systems can directly engage with their environment and execute complex user intent. Moreover, action systems can support agents to utilize available tools from external environments, thus significantly extending agents' task scopes. Therefore, the design of action systems will also determine the capability of AI agents in perception, decision making, execution, tool utilization, and any other components to align with the human brain. In other words, foundation models lay the groundwork for agents while action systems determine their ultimate potential to achieve complex targets. Designing an effective and comprehensive action system for AI agents is a critical endeavor that involves significant challenges and notable benefits. In Figure 8.1, we demonstrate the execution process of the action system in the cognition system. In this section, we will first discuss the human action system in Section 8.1, and then examine the transition from human action to agentic action in AI agents in Section 8.2. After that, we will systematically summarize the paradigms of existing action systems in AI agents, including action space, action learning, and tool learning, in Section 8.3. In Section 8.4, we analyze the differences between action and perception, and finally we summarize the conclusion in Section 8.5.

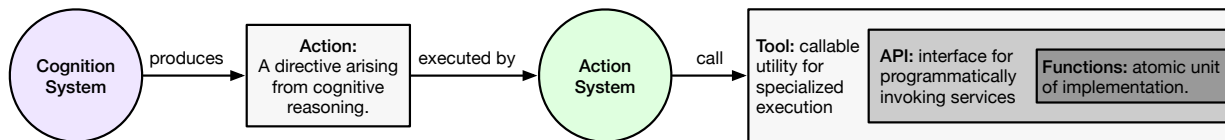


Figure 8.1: Illustration of several concepts related to action and action execution.

8.1 The Human Action System

Action system in human cognition refers to the processes that allow humans to perceive, plan, and execute goal-directed actions. It is a complex system that enables individuals to interact with a dynamic environment, make decisions, and adapt their behavior based on feedback. Generally, the action system within human cognition could be broadly categorized as *mental action* and *physical action*:

- **Mental action** can be viewed as a kind of distinct action, which is formulated as a thinking process to drive the final intention in the human brain. For example, reasoning, decision making, imagining, and planning can all be considered as various types of mental action. In other words, mental actions are equal to a brain signal that drives the physical actions of humans to fulfill the final objective.

- **Physical action** refers to any goal-directed bodily movement executed by the human motor system. To some extent, physical actions are usually expressed as a kind of continuous action. For example, speaking, manipulating, drawing, running, and grasping can all be regarded as physical actions. Employing a sequence of physical actions, humans can conduct the interaction and collect feedback from real-world environments.

Figure 8.2 illustrates a simple taxonomy of the human action system from the perspective of mental action and physical action. Empowered with both mental and physical actions, the human cognition system can handle diverse complex tasks from real-world scenarios. Drawing inspiration from human cognition, it is also essential for us to revisit how to formulate action systems in AI agents across different tasks, from language to digital and then in physical environments.

Model	Examples	Inputs	Objective	Definition
Large Language Model (LLM)	GPT-4 [7]	Language	Next-Token Prediction	LLM is to generate text based on the provided user prompts.
Large Multimodal Model (LMM)	LLaVA [513]	Multi-modal	Multi-modal Generation	LMM is to generate multimodal data based on multimodal inputs.
Robotic Foundation Model (RFM)	RT-1 [522]	Sensory inputs	Robotic Control	RFM is to generate robotic control based on the sensory inputs from dynamic environments.
Large Action Model (LAM)	LAM [622]	Interactive Environment	Executable Action	LAM is to generate executable actions based on the interactions within the environment.

Table 8.1: Definitions between different kinds of foundation models.

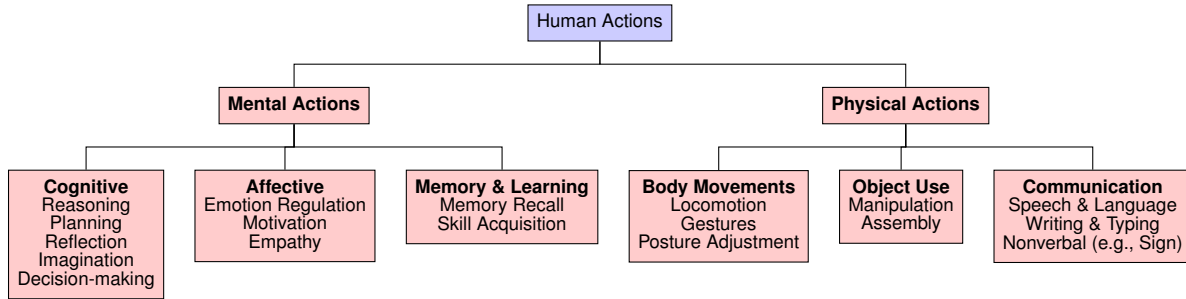


Figure 8.2: Illustrative Taxonomy of Human Actions, showing both mental and physical facets.

8.2 From Human Action to Agentic Action

In the past long period of time, human action systems [623] have significantly motivated us to shape the development of a computer system toward autonomous paradigms. The action mechanism plays a critical role in the human brain in driving goal-directed behavior. In an intelligent human brain [624], conscious and unconscious thinking signals are produced, converted into mental signals, which eventually lead to a sequence of action operations. This process can be mapped as a multi-stage pipeline that involves constructing action spaces, formulating learning mechanisms for improved decision making, and integrating external states (e.g., tools). Inspired by these principles, we discover that these designs are essential to formulate the prototype of AI agent.

Many existing frameworks incorporate action learning into their design or utilize it as an output. To clarify the definition of an action system, we highlight the distinctions among various frameworks, including large language models (LLM), large multi-modal models (LMM), robotic foundation models (RFM), and large action models (LAM), as shown in Table 8.1. Specifically, an LLM is to produce language output based on provided prompts, while an LMM is to generate multi-modality artifacts based on the multi-modal inputs. Existing language-based or digital AI agent frameworks are built upon these foundation models (e.g., LLM or LMM) via predefining the scope of action space and its learning strategies. On the other hand, an RFM is to optimize robotic control based on real-world environments (e.g., robotic video). Existing RFMs are pre-trained from web-scale video data and use video prediction to simulate the action of robotic control. The core of RFM is still to use the generative objective to learn knowledge from large-scale data, although it has involved some action designs in building physical AI agents. Moreover, some recent works [622] introduce the concept of large action model (LAM), which further highlights the stage to generate the action strategies, interact with real-world environments and enhance self-learning paradigm. From these definitions, we notice that, regardless of the foundational models employed, the core of action system is to build the interaction with the environment and then enable the learning process from the collected action trajectories via pre-defined reward

functions. Specifically, the mechanisms underlying these behaviors are also similar to the action system in human cognition, offering valuable insights for designing action systems in AI agent frameworks. For example:

- When processing different scenarios, humans usually will pre-define the action space to perform action trajectories to solve specific tasks. For instance, when playing computer games like Minecraft, we will set our action operations via keyboard or mouse to simulate behaviors like building house, mining gold, and so on. On the basis of this, we also need to build or create an action space for handling complex tasks in AI Agent frameworks.
- Compared to machines, the human cognitive system excels in continuously acquiring new knowledge through real-world interactions, guided by generating and optimizing the action sequences. Thus, replicating this learning ability in AI agents is essential to adapt the dynamic environment and build a new skill library.
- In addition, with the development of human civilization, learning to use external tools has been recognized as one of the most significant milestones in the evolution of human intelligence. By leveraging these external tools, humans can extremely extend the problem-solving capability in different scenarios, from the stone age to the industrial revolution.

To this end, we expect to build the mapping between the action system of human cognition system and the design of AI Agent framework, including how to build action space for AI agent from specific scenarios to general domain, how to build action learning within the environment, and how to leverage external states (e.g., tools) to extend the task scope of AI Agent. By developing this a systematic survey, we strive to provide more in-depth insights for the community with a clear understanding of the significance of action systems in AI agent frameworks.

8.3 Paradigms of Agentic Action System

Generally, the action system of AI agent frameworks consists of three major components: 1) the action space \mathcal{A} , which includes all types of action that agent can perform in real-world scenarios or downstream tasks, and can vary significantly depending on different agent settings, ranging from language-based agents to embodied agents; 2) the action learning within an dynamic environment that determines the state \mathcal{S} , observation \mathcal{O} and the optimization process of agent; 3) the tool space \mathcal{T} that encompasses the instruments, interfaces, or middle-wares the agent can perform for utilization, which ranges from physical devices such as robotic arms to digital interfaces like APIs. Overall, these components collectively define the scope and characteristics of the action system for AI agents, shaping their formulation and execution.

To fully explore the possible actions a_t in practical scenarios, we must formally represent the action space and consider both individual operations and the underlying hierarchical reasoning processes. This means examining the action space at various levels, from low-level manipulations to high-level operators that orchestrate complex workflows.

Accordingly, the AI agent decision making process can be formalized as a trajectory $\langle o_t, s_t, a_t \rangle$, where a_t is selected from the action space \mathcal{A} to transform the current state s_t based on observation o_t into the next state. In some cases, integrating external tool systems may also be necessary. By executing a sequence of $\langle o_t, s_t, a_t \rangle$, the agent is steered toward achieving its final objectives.

8.3.1 Action Space Paradigm

Action space \mathcal{A} is an important component, which serves as the basis for building an action system within AI agent frameworks. The composition of the action space determines how AI agents solve complex tasks in different scenarios. In Figure 8.2, we present an illustrative taxonomy of the action system based on its action space. Generally, we summarize the action space within existing works as three distinct types, as outlined below.

Language Language-based AI agents typically operate through language-driven actions in interactive linguistic environments, such as reasoning, programming, retrieving information, executing API calls, or interacting with external tools. In our study, we summarize three distinct types of language-based action spaces, including plain text, code programming, and communication. Specifically, early language-based AI agents are built with plain text, which aim to perform interactive decision-making in verbal environments or text-based games. Here, ReAct [70] is a representative language-based AI agent, which synergizes the reasoning and actions of an LLM to solve various problems. AutoGPT [625] analyzes and decomposes user requests into multiple subtasks and uses web search or other tools to tackle each of them. Reflexion [48] involves self-refinement and the memory mechanism to enhance action execution in language tasks. LLM+P [163] empowers LLM-based agent with planning capability to aid decision-making. However, converting plain text into an executable command usually requires LLMs to first interpret the text and then perform instruction conversion, leading to additional information loss. To this end, some work explores using

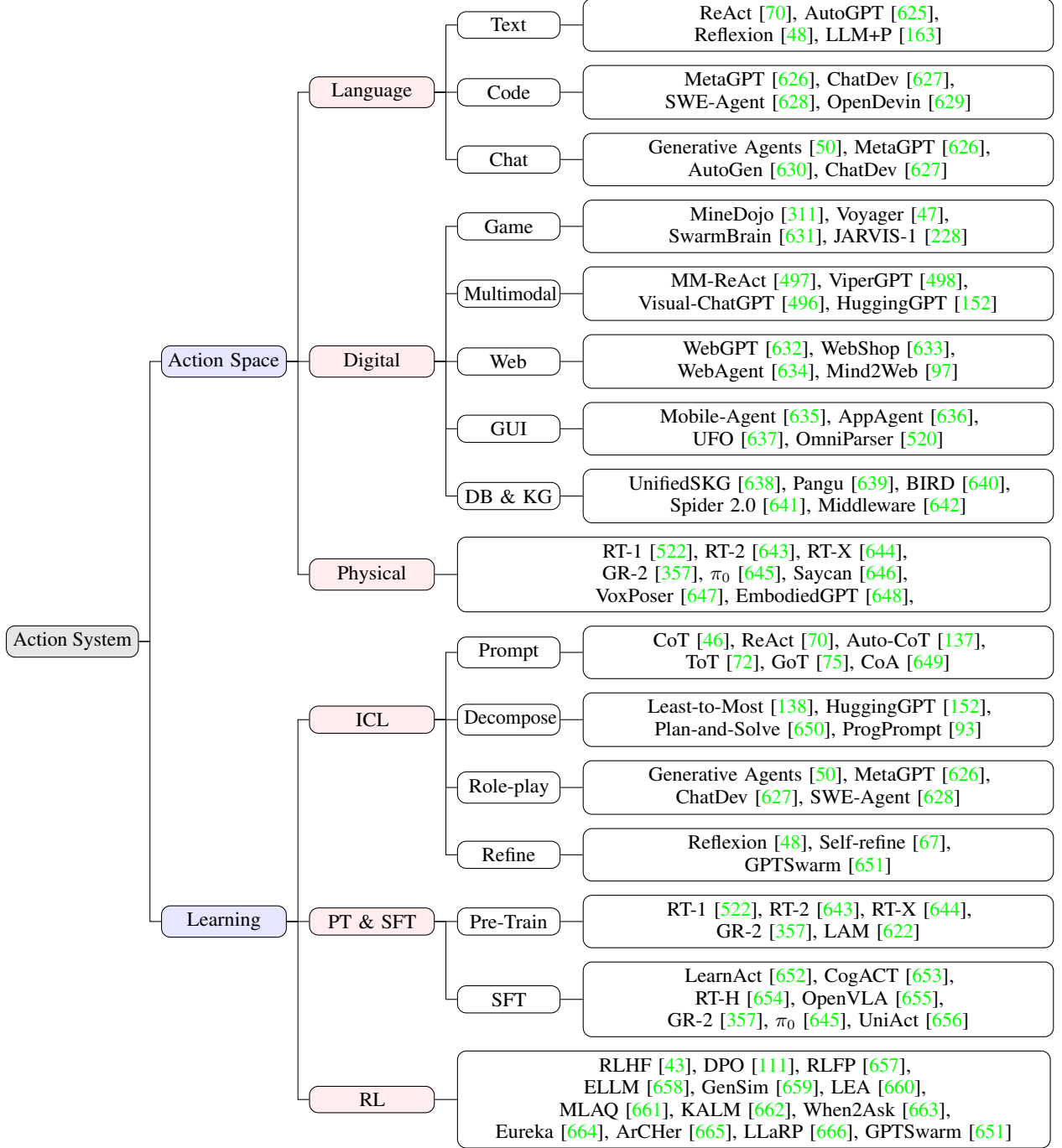


Figure 8.3: Illustrative Taxonomy of Action system, including action space and learning paradigm.

code as the action space, allowing direct execution of the generated code and self-verification. MetaGPT [626] and ChatDev [627] build the action space via programming language with multi-agent collaboration. SWE-Agent [628] consider different stages of software engineering and thus solve software issues. OpenDevin [629] devises an automatic software development platform that integrate code writing, interaction with the command, sandbox for code execution, and collaborations. Moreover, some frameworks are built based on multi-agent communications, and then use chatting to analyze which actions should be employed in the next step. Here, Generative Agents [50] directly simulate multiple characters in a virtual town, to explore how each agent to conduct next action. MetaGPT [626] and ChatDev [627]

are both multi-agent frameworks to facilitate the development of software engineering. AutoGen [630] is also a representative framework that enable multiple agent collaboration to solve any complex tasks. Generally, language-based AI agents, empowered by LLMs, perform effectively in linguistic interactions. However, limited to the scope of the action space, it also poses challenges of how to solve more complex tasks in real-world scenarios. Therefore, we also need to formulate new research solutions to construct a more sophisticated action space to solve challenging tasks.

Digital To expand the capabilities of AI agents beyond language, some works have also developed advanced AI agents that operate within digital environments, such as web proxies, online shopping platforms, and gaming systems. For examples, MineDojo [311] devises a virtual agent via video-language pre-training and simulates an environment that supports a multitude of tasks and goals within Minecraft. Moreover, Voyager [47] is an embodied AI agent trained to play Minecraft. It simulates multiple executable actions in code form to develop a skill library via interacting with the Minecraft environment, and thus improve the capability of virtual agents. JARVIS-1 [228] is an open-world agent that can handle multi-modal inputs / outputs, generate sophisticated plans, and perform embodied control. It explores the evolutionary behaviors of the agent when acting in Minecraft. SwarmBrain [631] is an embodied agent that uses LLMs to act strategically and in real time in StarCraft II. Additionally, some research studies investigate how LLMs can act to process multimodal tasks. MM-ReAct [497] and ViperGPT [498] apply LLMs to perform the thinking process for multimodal tasks and then select visual experts for task solving. Visual-ChatGPT [496] integrates multiple visual experts and uses LLMs as the controller to solve tasks. HuggingGPT [152] directly involves four stages, including task planning, model selection, model execution and response generation, to automatically analyze user instructions and predict the final answers based on complex multimodal tasks. It is also vital for the agent to keep up with the latest information available online. Therefore, some AI Agent frameworks (e.g., WebGPT [632], WebAgent [634]) are designed to interact with search engine to enhance the capability of agent to discover the answers from website. WebShop [633] is used to explore the potential of AI Agent for online shopping. Mind2Web [97] is to build a generalist agent that simulate multiple complex web tasks. As foundation agents advance in processing multimodal tasks or web tasks, there is a increasing trend to enhance their capability in solving complex computer tasks. Mobile-Agent [635] utilizes multimodal models as the cognitive controller to manage and orchestrate mobile functionalities. AppAgent [636] defines various app usages as action spaces, enabling foundation models to interact with different apps as a mobile intelligent assistant. UFO [637] and OmniParser [520] are two advanced GUI agents which manipulates UI operations as the action space, enabling AI agent to perform computer-use tasks. Generally, empowered with more advanced skills in digital environment, AI agent can demonstrate better intelligent in solving complex tasks, and represent a significant shift from language intelligent to digital intelligent. By expanding the action space to include web browsing, GUI interaction, mobile applications, and embodied systems, AI agents are evolving into more autonomous, multimodal, and context-aware systems, bridging the gap between foundation models and human cognition systems. In addition, other research explores LLM integration with structured digital environments such as relational databases and knowledge graphs (KGs). Pangu [639] pioneered the connection between LLMs and large-scale KGs, while BIRD [640] and Spider 2.0 [641] established a foundation for LLMs to operate with enterprise databases in real-world settings. NL2SQL-BUGs [667] addresses the critical challenge of identifying semantic errors in NL2SQL pipelines [365], which enhances the reliability of LLM-driven interactions with relational databases [668]. Similarly, frameworks like UnifiedSKG [638] and Middleware [642] expand LLMs’ action capabilities across both databases and KGs.

Physical Building an AI agent to interact with the real physical world can be viewed as the ultimate objective to simulate a computer program to act as a human cognition system. To achieve this, we require the agent to be capable of processing signals from real-world environments and generating feedback to facilitate continuous improvement. Therefore, it will pose new challenges on how to process the continuous signals collected by sensors and enable foundation models to make decisions. To fulfill this, RT-family [522, 643, 644] pre-trained vision-language-action models to integrate knowledge from web videos into robotic learning, enhancing robotic control and action execution. GR-2 [357] is a robotic model that undergoes large-scale pre-training on video clips and language data, followed by fine-tuning on robot trajectories for robotic action prediction. π_0 [645] pre-trained a robotic model based on robot platforms, including single-arm robots, dual-arm robots, and mobile manipulators, to build robotic learning in physical systems. SayCan [646] bridges the connections between robotic semantics and LLMs, using the robotic model to provide perception for LLMs and then using LLMs to make high-level decision-making. VoxPoser [647] uses LLMs to understand and decompose 3D Value Maps for Robotic Manipulation. Besides, EmbodiedGPT [648] utilizes vision-language models to understand video data and perform decision-driven actions. In physical environments, it is worth noting that we usually need to understand continuous signals and then generate continuous actions for robotic control. Despite the existing foundation models that can effectively process discrete-level actions (e.g., language or computer-use), how to process long continuous signals is still challenging. Therefore, eliminating the differences between continuous signals and discrete signals in foundation models is still a major problem.

Generally, action space serves as one of the most critical components in building an effective AI Agent system. An effective action space enhances the capability and efficiency of the AI Agent in processing downstream tasks. Action space usually ranges from the discrete space (e.g., skill library in Atari games) to the continuous space (e.g., robotic manipulation). As AI agents become more autonomous and multimodal, designing effective action spaces will be crucial for advancing general-purpose AI systems capable of real-world interactions.

8.3.2 Action Learning Paradigm

In the human cognition system, action learning [669] represents the problem-solving process, involving both taking actions and reflecting on feedback. Similarly, action learning for AI agents refers to the iterative process by which an autonomous AI system refines its decision making and behavior through direct interaction with the real world environment. Generally, action learning encompasses a cycle of multiple stages, including building action space, choosing actions, and optimizing action selection based on interaction with the environment (e.g., receiving feedback or rewards and adjusting policy for choosing actions). By iteratively deploying these strategies, AI agents can adapt to the latest information or changing conditions in real time, ultimately enabling more robust, flexible, and efficient problem-solving capabilities. Therefore, an effective action learning mechanism is crucial for the optimization of agentic action systems. In this part, we mainly focus on three different representative learning paradigms, including in-context learning, supervised training, and reinforcement learning, which are discussed below:

In-context Learning As large language models have demonstrated emergent ability, in-context learning has been considered as the most effective method to leverage the existing capabilities of LLM without any modifications. Provided with well-designed prompts to describe actions, AI agents can understand specific actions, perform these actions, reflect on the outcome of the interaction with the environment, and finally achieve goals. Among these approaches, the common method is to use prompting techniques to instruct LLMs to generate agentic action. Here, the most representative one is Chain-of-Thought (CoT) [46] prompting, which applies “*Let us think step by step*” technique to generate a sequence of intermediate reasoning steps, exploring potential solutions systematically. ReAct [70] enables LLMs to generate reasoning trails and task-specific actions through interaction within the environment, improving the reasoning and decision-making capabilities of AI agents. LearnAct [652] devises an iterative learning strategy to expand action space by generating code (i.e., Python) to create and revise new actions. Moreover, some works (e.g., Auto-CoT [137] explores how to automatically generate CoT via LLMs and then enable the autonomous thinking process of AI agents. To handle more complex tasks, ToT [72] considers the thought process as a tree structure and introduces the tree search via LLM prompting, while GoT [75] applies a graph structure along with the graph search. For robotic models, CoA [649] designed four different prompt settings (e.g., object, grasp, spatial, and movement) to allow robot manipulation with reasoning process. Furthermore, to tackle more complex tasks that require intricate agentic workflows, some frameworks introduce the stage of task decomposition via LLM prompting to break down user instructions. Least-to-Most [138] is a classical prompting technique to convert user instructions into multiple subtasks. HuggingGPT [152] is a representative AI agent framework that applies task planning to transform user requirements into actionable items. Plan-and-Solve [650] directly uses LLM to make plans from user instructions and then give answers based on the generated plans. Progprompt [93] applies similar task decomposition to robotic tasks. In addition, using prompting techniques to formulate the characteristic of AI agent has also been considered as an increasing trend to facilitate the simulation and productivity of AI agent frameworks (e.g., Generative Agents [50], MetaGPT [626], ChatDev [627], SWE-Agent [628]). Finally, some other frameworks (e.g., Reflexion [48] or Self-refine [67]) analyze the external feedbacks of user interaction within the environment and then iteratively refine and polish results via well-designed reflexion prompts. All of these designs allow us to better understand user instructions, decompose task goals, and make plans for thinking answers. In-context learning can help us avoid parameter optimization and reduce the heavy cost of training LLMs. It allows AI agents to perform various actions effectively and adapt to a wide range of domains. However, challenges still remain if we want to acquire agents of even stronger action learning ability.

Supervised Training To further improve the action learning ability of foundation models, increasing research efforts have focused on training methodologies, including self-supervised pretraining (PT) and supervised fine-tuning (SFT). For the pre-training paradigm, the most representative works is RT-family [522, 643, 644], which pre-trains robotic Transformer on large-scale web and robotic data, yielding a powerful vision-language-action model. Following this policy, GR-2 [357] is developed through extensive pre-training on a large corpus of web videos to understand the dynamics of the world and post-training on robotic trajectory data to specialize in video generation and action prediction. Similarly, LAM [622] is a large action model pre-trained on trajectories of user interaction with computer usage. However, the pre-training paradigm usually incurs massive computation costs. Therefore, many works take the fine-tuning paradigm to enhance the action capability of foundation models. OpenVLA [670] is built upon the Llama2 [11] language model and incorporates a visual encoder based on DINOv2 [671] and SigLIP [672]. It is fine-tuned on a diverse set of real-world robot demonstrations from Open X-Embodiment (OXE) [673] and outperforms RT-2-X [673]

across different tasks, all while utilizing $7\times$ fewer parameters. Building upon OpenVLA, CogACT [653] integrates an additional diffusion action module and introduces an adaptive action ensemble strategy for inference. It is also fine-tuned using datasets from OXE and demonstrates a 35% improvement in the SIMPLER [674] simulated environment and a 55% increment in real robot tasks using the Franka Arm. Besides, some works also explore how to enable robotic model to learn action from plain language in physical world. For examples, RT-H [654] introduces a hierarchical architecture to build action space, which first predict language motions and then generate low-level actions. And π_0 [645] collected massive diverse datasets from different dexterous robot platforms, and then fine-tune the pre-trained VLMs to learn robotic actions. UniAct [656] learns universal actions that capture generic atomic behaviors across differently shaped robots by learning their shared structural features. This approach achieves cross-domain data utilization and enables cross-embodiment generalizations by eliminating heterogeneity [132]. Overall, using supervised training, including pre-training and supervised fine-tuning, can effectively adapt foundation models to perform actions intelligently in real-world scenarios. Last but not least, it is worth noting that, even with extensive training on a vast corpus, it is still beneficial to apply in-context learning on top of the trained model for AI agents, in an pursuit for their best performance.

Reinforcement Learning To facilitate an action learning procedure in addition to in-context learning and supervised training, it is also crucial for agents to interact with the environment and eventually optimize their action policy through experience, feedback, or rewards. Considering this iterative and sequential nature, reinforcement learning (RL) provides us with the systematic methodology we need [675, 676, 677, 678]. In RL paradigms, there are several classical and representative algorithms, such as Deep Q-Network (DQN) [679] and Proximal Policy Optimization (PPO) [680]. The most representative RL work that applied reinforcement learning to foundation models is InstructGPT [43], which effectively aligns LLM outputs with human preferences via RLHF. Since RLHF usually requires additional training to build the reward model, some papers (e.g. DPO [111]) proposes to directly optimize preference data through contrastive learning. Existing work [89, 681] also demonstrate the potential of scaling the RL algorithm for foundation models to produce long CoT thinking stages with impressive performance. Although RL paradigms have been successfully used to fine-tune LLMs for text generation tasks [12, 682, 43, 683], efficiently utilizing the RL algorithm for action learning remains one of the many challenges that require further attempts. Recent advances indicate significant progress in applying RL to action learning with LLMs from various perspectives:

- Given the rich world knowledge encapsulated in LLM, we can use LLM to *mimic external environments or generate imagined trajectories* to aid agents in action learning. For instance, RLFP [657] utilizes guidance and feedback from the policy, value, and success-reward foundation models to enable agents to explore more efficiently. Similarly, ELLM [658] utilizes large-scale background knowledge from LLMs to guide agents in efficient exploration within various environments. GenSim [659] automatically generates rich simulation environments and expert demonstrations by exploiting the coding abilities of LLM, thereby facilitating the capability of the agent for free exploration. LEA [660] leverages the language understanding capabilities of LLM and adapts LLM as a state transition model and a reward function to improve the performance of offline RL-based recommender systems. MLAQ [661] utilizes an LLM-based world model to generate imaginary interactions and then applies Q-learning [684] to derive optimal policies from this imaginary memory. KALM [662] fine-tunes LLM to perform bidirectional translations between textual goals and rollouts, allowing agents to extract knowledge from LLM in the form of imaginary rollouts through offline RL. In general, empowered by RL paradigms, we can significantly explore the internal knowledge from LLMs and thus enhance the interactions with external environments. Current works such as Search-R1 [685], R1-Searcher [686], RAGEN [687], and OpenManus-RL [688] are exploring utilizing RL methods to fine-tune the agent models on trajectory data in agentic environments.
- Besides, hierarchical RL is also a promising topic that helps foundation model to decompose complex task and then learn optimal policies to solve each task via RL paradigm. For example, When2Ask [663] enables agents to request high-level instructions from LLM. The high-level LLM planner provides a plan of options, and the agent learns the low-level policy based on these options. Eureka [664] leverages LLM to generate human-level reward functions with reflection, allowing agents to efficiently learn complex tasks such as anthropomorphic five-finger manipulation. ArCher [665] adopts a hierarchical RL approach, utilizing an off-policy RL algorithm to learn high-level value functions, which in turn implicitly guide the low-level policy. LLARP [666] leverages LLM to comprehend both textual task goals and visual observations. It employs an additional action output module to convert the output of the LLM backbone into a distribution over the action space. Overall, using hierarchical RL can guide AI Agent to explore optimal strategies when analyzing user requests for reasoning and planning.

Using reinforcement learning, we can integrate foundation models with online learning from interactive environments, incorporating both action policies and world models. This integration enables advanced action systems in AI agents. Within the reinforcement learning paradigm, agents dynamically adapt and refine their decision-making processes in

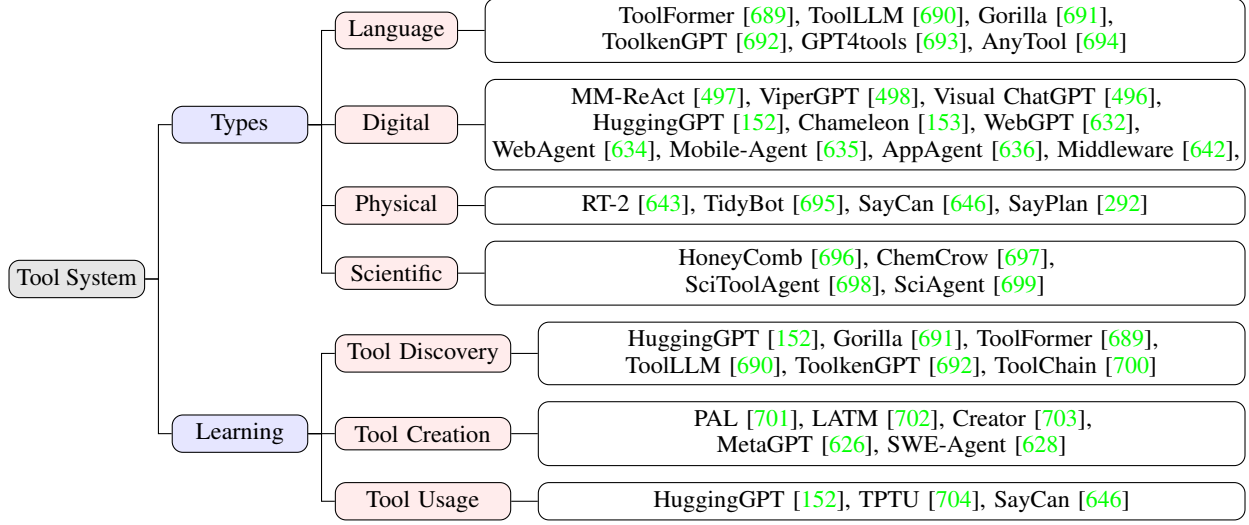


Figure 8.4: Illustrative Taxonomy of Tool Systems in AI Agents, including tool category and learning paradigm.

response to external feedback, facilitating greater efficiency and effectiveness in action learning and achieving desired outcomes.

Summary In general, Empowered by action systems, AI agents have demonstrated significant decision-making capabilities across various fields. For example, action learning enables AI agents to automate the understanding of Graphical User Interfaces (GUIs) and perform various operations, thereby improving human productivity through automatic computer usage. Moreover, several studies have shown that AI agents equipped with action systems can achieve remarkable outcomes in robotic manipulation tasks, such as object picking, laundry folding, and table cleaning. There are also promising research directions in the industry employing action models. For instance, autonomous driving (AD) has attracted considerable attention due to the exceptional performance of VLMs in perception and decision-making. By integrating human understanding through foundation models, AD systems can effectively comprehend real-world surrounding, enabling them to simulate human-level drivers. In summary, action learning endows agents with the ability to interact with the external world, thereby creating more opportunities for AI applications in real-world scenarios.

8.3.3 Tool-Based Action Paradigm

Tool learning distinguishes human intelligence from that of other animals. Ever since the Stone Age, human use of tools has boosted efficiency, productivity, and innovation. Similarly, enabling AI agents to operate in digital and physical environments by harnessing various tools is a fundamental step toward achieving human-level intelligence.

Definitions In AI, tools are defined as interfaces, instruments, or resources that allow agents to interact with the external world. Examples include web search [632, 705, 97, 634], databases [706, 707, 708, 709], coding environments [710], data systems [711, 712, 713], and weather forecasting [714]. By translating tool functionality into plain text or API formats, foundation models can expand their problem-solving scope. The evolution of tool systems in AI can be summarized in stages. Initially, with the advent of large language models [2], the focus was on converting tools into explainable formats (e.g., function calls). Later, advances in multimodal processing shifted interactions from conversational chats to graphical user interfaces (GUIs), and more recent work has explored embodied agents that control hardware (e.g. robotic arms, sensors) to interact with the physical world. To simplify, a tool-based action can be considered a form of external action employed for assistance.

Tool Category Similar to action spaces, tools can also be classified into multiple categories according to their types. In this part, we mainly summarize three key domains, including language, digital, and physical. In addition, we also explore the potential of tool learning in emerging areas such as scientific discovery:

- *Language*: To facilitate the use of external tools, we usually denote the tool as a kind of function call for foundation models, which usually encompasses task descriptions, tool parameters, and corresponding

outputs. This expression allows LLMs to understand when and how to use tools in AI agents. Specifically, ToolFormer [689] expands the capabilities of language models by integrating external tool spaces, including calculator, QA systems, search engine, translation, and calendar. ToolLLM [690] uses RapidAPI as the action space and then uses a depth-first search-based decision tree algorithm to determine the most suitable tool for solving tasks. Gorilla [691] is a fine-tuned LLM based on the tool documents and then can be used to write API calls. ToolkenGPT [692] is to optimize tool embeddings and then enable LLMs to retrieve tools from the fine-tuned tool embeddings. GPT4tools [693] and AnyTool [694] are also building self-instruct datasets and then fine-tune LLMs on them for tool usage. Generally, due to the impressive capability of LLMs, language-based tool utilization for AI agents has been studied, with its effectiveness validated in abundant works, ranging from plain text or function calls to code programming.

- *Digital*: With the success of LLMs in processing language information, many researchers are exploring extending the task scope of AI agents from the language to the digital domains (e.g., MultiModal, Web search, GUI, and so on). For example, MM-ReAct [497], ViperGPT [498], and Visual ChatGPT [496] employed LLMs as the controller and then used LLMs to select visual experts for solving different tasks. HuggingGPT [152] and Chameleon [153] use LLMs to first conduct reasoning and planning actions and then analyze which multimodal tools should be used for solving user instructions. WebGPT [632] and WebAgent [634] respectively empowered LLMs with search engines to enhance the capability of LLMs to solve more challenging tasks. Mobile-Agent [635] and AppAgent [636] respectively incorporate GUI manipulations and App usage as the tool-based actions to extend the task scope of AI agents in solving mobile phone tasks. In contrast to the physical world, digital environments usually provide simpler pipelines to collect and process data. By involving foundation models and their interaction with the digital environment, it is possible for us to develop intelligent assistants in computers, mobile phones, and other digital devices.
- *Physical*: For physical world applications, RT-2 [643] demonstrates language-guided robotic manipulation using visual-language tools, and TidyBot [695] shows how LLMs adapt cleaning tools to personalized household preferences. SayCan [646] uses LLMs as the cognitive system to guide robots in solving tasks through robotic arms and visual perception. SayPlan [292] built a 3D scene graph as the action spaces and designed multiple actions and tools for 3D simulation, and then used LLMs as planners to invoke these actions or tools for robot task planning. Besides, specialized applications in real-world scenarios now also proliferate across different domains. For instance, in surgical robotics, [715] presents a multi-modal LLM framework for robot-assisted blood suction that couples high-level task reasoning, enabling autonomous surgical sub-tasks. Some autonomous driving systems [716, 717] also integrate vision-language models with vehicle control tools for explainable navigation. In total, physical world applications pose the most significant challenge when compared to other tasks, but they also offer the biggest industrial value. Therefore, it still requires us to continue exploring advanced action learning and tool integration in physical-based agents in the future.
- *Scientific*: Scientific tools have played a transformative role in advancing AI agents across disciplines, enabling them to learn, adapt, and execute tasks while integrating foundational models with frameworks that drive innovation and address complex challenges. In materials science, HoneyComb [696] exemplifies tool-driven advancements with its ToolHub. General Tools provide dynamic access to real-time information and the latest publications, effectively bridging gaps in static knowledge bases. Material Science Tools are designed for computationally intensive tasks, leveraging a Python REPL environment to dynamically generate and execute code for precise numerical analysis. Similarly, ChemCrow [697] demonstrates the transformative power of tools in chemistry by integrating GPT-4 with 18 expert-designed tools to automate complex tasks such as organic synthesis, drug discovery, and materials design. These tools include OPSIN for IUPAC-to-structure conversion, calculators for precise numerical computations, and other specialized chemistry software that enables accurate reaction predictions and molecular property evaluations. Similarly, SciToolAgent [698] showcases how multi-tool integration can revolutionize scientific research. Designed to address the limitations of existing systems, SciToolAgent integrates over 500 tools (e.g., Web API, ML models, function calls, databases, and so on). Finally, SciAgent [699] exemplifies a multi-agent framework that integrates ontological knowledge graphs with specialized agents for hypothesis generation and critical analysis, emphasizing the power of modular, tool-driven systems to accelerate discovery in materials science and beyond. These examples underscore the transformative potential of integrating specialized tools into AI frameworks to address domain-specific challenges effectively.

Tool learning Inspired by human evolution [718], the integration of tools in AI involves three key aspects: *Tool Discovery* (identifying suitable tools), *Tool Creation* (developing new tools) and *Tool Usage* (effectively employing tools). We also systematically review existing literature and summarize them in the following:

1. **Tool Discovery:** In real-world environments, there is a wide range of tools from the digital to the physical world. Finding the most appropriate tools for user instructions can be challenging. Therefore, the process of tool discovery is to identify and select the appropriate tools that AI agents can operate on to achieve their objectives. This stage also requires the world models in AI agents to have a profound understanding of any complex user instructions and world knowledge of different tools. Moreover, the versatility of AI agents is also correlated with its ability to operate diverse tool systems. Generally, tool discovery can be categorized into two mainstream paradigms: retrieval-based and generative-based methods. Retrieval-based methods aim to select the most relevant tools from the tool library. For example, HuggingGPT [152] introduces a framework in which LLMs act as controllers, orchestrating task planning and then invoking suitable models from platforms such as Hugging Face to fulfill user intention. In generative-based approaches, we often fine-tune LLMs to learn how to use and select tools based on various user instructions. For instance, ToolFormer [689] collects a massive corpus with the corresponding API calls (e.g., calculator, QA system, search engines, translation, and calendar) for training. ToolLLM [690] collect tool instructions based on solution paths and then fine-tune Llama models to generate better API calls for tool utilization.
2. **Tool Creation** In addition to using existing tools, the ability to create new tools plays a crucial role in human civilization. For language agents, a widely adopted approach is to use LLMs to generate functions as executable programs, which consist of both the code and documentation. For example, PAL [701] generates programs as intermediate reasoning steps to solve problems, LATM [702] or Creator [703] use LLMs to create code for user intentions, and to further design a verifier to validate the created tools. SciAgent [699] not only integrates multiple scientific tools but also crafts new tools for scientific discovery. More details on tool creation from an optimization perspective can be found in Section 9.4.2.
3. **Tool Usage** After collecting or creating tools, the effective use of tools constitutes the cornerstone of the capabilities of AI agents, allowing applications that bridge virtual and physical worlds. Modern AI agents increasingly employ tools to tackle complex tasks across diverse domains, with three key dimensions of expansion: 1) *Vertical Specialization*: Agents leverage domain-specific tools to achieve professional-grade performance in complex fields such as robotics, science, and healthcare; 2) *Horizontal Integration*: Systems combine multiple toolkits across modalities (vision, language, control) for multimodal problem-solving; 3) *Embodiment*: Agents physically interact with environments through robotic tools and sensors.

Summary Tool learning and action learning constitute the two most important components of the action system in AI agents. Tool learning can be considered as a kind of action to use external states for problem-solving. Tool learning enables AI agents to substantially broaden their range of tasks, pushing the boundaries beyond the scope of foundation models. For example, empowered by API or function calls, language models can directly reuse the capability of existing models (e.g., retrieval, coding, web search) to generate answers, rather than next-token prediction [719]. Tool learning also involves multiple challenging stages, including how to determine the tool space, how to discover and select tools, and how to create and use tools. Overall, tool learning plays a pivotal role in building an omnipotent AI agent framework to solve complex tasks in different domains.

8.4 Action and Perception: “Outside-In” or “Inside-out”

A central debate in cognitive science and neuroscience concerns whether action or perception stands at the root of causal flow in intelligent systems. Figure 8.5 presents different perspectives. The traditional “outside-in” view insists that causal influence begins with external stimuli. The environment excites peripheral receptors, these signals propagate inward, and eventually produce behavior. This perspective portrays the organism—or agent—as essentially reactive: the external world causes sensory changes, and the agent’s actions represent a downstream effect of those changes. In contrast, Buzsáki’s “inside-out” framework [18] proposes that it is the agent’s own actions that shape the meaning and consequences of incoming signals. Such a view implies an active agent, one which continuously generates predictions and motor commands, while sending “corollary discharges” or “action copies” to sensory areas. These internally generated signals serve as references that inform the agent which sensory changes are self-initiated rather than imposed by the outside world. In this manner, cause shifts from an external event to an internally launched initiative, leaving external stimuli to play a confirmatory or corrective role. This reversal has significant implications for how we interpret perception’s purpose and function: it is not an end in itself, but a means of updating and refining the agent’s own action-driven hypotheses about the environment.

From an evolutionary perspective, possessing the ability to move without relying on sophisticated sensory analysis can yield immediate survival benefits. Even simple organisms profit from periodic motion that stirs up food in nutrient-rich water, long before elaborate perceptual capacities evolve. In other words, movement precedes advanced sensing in evolutionary time, suggesting that the capacity to act is not merely the effect of external stimuli but can

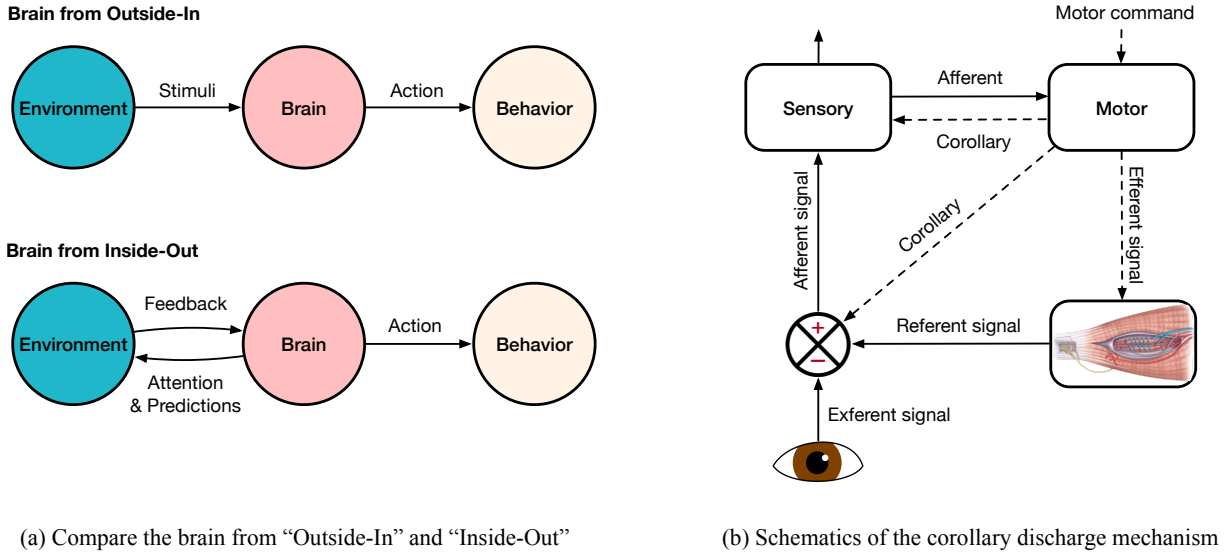


Figure 8.5: (a) Compare the brain from “outside-in” and “inside-out”. (b) Illustration of the schematic of the corollary discharge mechanism. A motor command (efferent signal) travels from motor areas to the eye muscles, while a corollary discharge (dashed arrow) is routed to a comparator in the sensory system. The comparator uses this internal signal to modulate or subtract external (exafferent) input. Additionally, tension feedback from the muscles (reafferent signal) exerts a delayed effect on perception. Direct projections from motor to sensory cortices underlie this architecture in all mammals. Part (b) is adapted from the original figure in [18].

itself be the driving cause of subsequent perceptual development. It is precisely when action mechanisms become sufficiently established that the agent benefits from additional sensors, which guide those movements more strategically. This developmental sequence grounds perception in utility, tying sensory discrimination to the practical outcomes of movement.

Disruptions in the normal interplay of action and perception illuminate the intricate cause-effect loop. During sleep paralysis, the brain’s motor commands temporarily fail to reach the muscles; external stimuli still bombard the senses, but the usual action-to-perception calibration is lost. As a result, the individual experiences a heightened sense of unreality because the brain lacks internally generated reference signals to interpret sensory input. Similarly, if one externally manipulates the eye without the brain issuing a motor command, the visual scene appears to move, highlighting how perception alone—devoid of a preceding, self-initiated action—risks confusion. Neurophysiological data further support the inside-out model. Many neurons in areas once deemed “purely sensory” track not only changes in external stimuli but also self-generated movements—sometimes more strongly so. This indicates that “cause” in the brain frequently emerges from within, guiding both the magnitude and meaning of external signals. Without these internal correlates, raw sensory data can become ambiguous or even useless to the system.

Implications for Intelligent Agents The inside-out perspective offers potent insights for modern research on intelligent agents. Most contemporary AI systems—and many LLM agents—still function predominantly in a reactive mode, awaiting user input and generating responses based on statistical correlations learned from vast datasets. Such passivity resembles an “outside-in” framework, where the agent’s role is limited to responding, not initiating. Yet if an agent were to be active, continuously forming and testing hypotheses via self-initiated behaviors (physical or representational), it might ground its own “perceptual” inputs—be they sensory streams or linguistic prompts—and thereby reduce ambiguity. For instance, an LLM-based agent that interjects questions or verifies its own statements against a knowledge base could better discern which inferences are self-caused from those demanded by external data. By tracking these self-initiated contributions (analogous to corollary discharge), the model could improve coherence, lessen errors known as “hallucinations”, and refine its internal state through iterative cause-effect loops.

A proactive stance also encourages more data-efficient and context-aware learning. Instead of passively waiting for labeled examples, an agent can explore, provoke feedback, and incorporate self-generated experiences into its training. Over time, this tight coupling between action and perception may bolster the agent’s ability to handle complex tasks, adapt to unanticipated challenges, and generalize more robustly. The shift from an outside-in to an inside-out model reframes perception as causally downstream of action. Intelligent systems—whether biological or artificial—stand

Table 8.2: Comparing the perception and action of human and AI agents.

Dimension	Human Brain / Cognition	LLM Agent	Remarks
Perception	<ul style="list-style-type: none"> - Integrates multiple sensory channels (vision, hearing, smell, touch, taste). - Perception closely tied to emotions, endocrine system, and physical state. - Highly sensitive, capable of detecting subtle differences. 	<ul style="list-style-type: none"> - Primarily language-based with some multimodal capabilities. - Perception depends on external sensors and models with limited integration. - Lacks real-time coupling with physical states. 	Perception differences lead to varying ways of understanding reality. Embodied AI attempts to bridge this gap but still faces both hardware and software challenges.
Unified Representation	<ul style="list-style-type: none"> - Simultaneously processes multimodal inputs: vision, hearing, language, motion, and emotions. - Different brain regions collaborate to create unified spatiotemporal and semantic understanding. 	<ul style="list-style-type: none"> - Primarily text-based. Some multimodal models can process images or audio but with low integration. - No fully unified spatiotemporal modeling like the human brain. 	Even advanced multimodal models lack the human brain’s holistic, unified representation capacity. Hardware and algorithmic challenges remain.
Granularity in Task Switching	<ul style="list-style-type: none"> - Flexible in shifting between macro and micro cognitive tasks. - Can plan at a high level and shift focus to finer details when needed. - Adjusts task priority and focus dynamically based on context and working memory. 	<ul style="list-style-type: none"> - Relies heavily on prompt engineering for granularity control. - Cannot autonomously reallocate attention between task layers. - May get stuck in a specific level of abstraction in absence of guided prompts. 	Humans can dynamically adjust cognitive granularity based on situational demands, while LLMs require explicit instruction to switch task focus effectively.
Action	<ul style="list-style-type: none"> - Goal-oriented process drives multiple sensory to make decisions. - Real-time Learning from the experience via the environmental interaction. - Encompass both physical activities and mental processes. 	<ul style="list-style-type: none"> - Action space need to be defined in advance. - Unable to support actions in continuous spaces. - Relies on online training to optimize the decision-making process in the environment. 	Humans are capable of actively learning new actions and performing continuous actions, whereas LLM agents currently lack this capability.

to benefit from recognizing that purposeful movement, or proactive conversational steps in the case of LLMs, can actively create, shape, and interpret the signals that flow back in. By acknowledging the cause-effect power of action and striving to build active rather than merely reactive agents, we may approach a deeper understanding of both natural cognition and the next generation of AI.

8.5 Summary and Discussion

Traditionally, action represents the behaviors of the human cognition system based on the interactive feedback from the environment. It endows humans with the capability to think, reason, speak, run, and perform any complex manipulations. Based on the action system, humans can iteratively evolve the brain intelligence by enhancing their perception and actions from the world, and form a closed loop to further create new civilization and innovation in the world. Similarly to a human cognition system, the action system plus the tool system also play an important role for AI agents. Integrating action systems allows AI agents to systematically plan, execute, and adjust their behaviors, facilitating more adaptable and robust performance in dynamic contexts. In this section, we systematically examine and summarize the impact of the action module on AI agents, focusing on both action systems and tool systems.